

*Commenced Publication in 1973*

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

## Editorial Board

David Hutchison

*Lancaster University, UK*

Takeo Kanade

*Carnegie Mellon University, Pittsburgh, PA, USA*

Josef Kittler

*University of Surrey, Guildford, UK*

Jon M. Kleinberg

*Cornell University, Ithaca, NY, USA*

Friedemann Mattern

*ETH Zurich, Switzerland*

John C. Mitchell

*Stanford University, CA, USA*

Moni Naor

*Weizmann Institute of Science, Rehovot, Israel*

Oscar Nierstrasz

*University of Bern, Switzerland*

C. Pandu Rangan

*Indian Institute of Technology, Madras, India*

Bernhard Steffen

*University of Dortmund, Germany*

Madhu Sudan

*Massachusetts Institute of Technology, MA, USA*

Demetri Terzopoulos

*New York University, NY, USA*

Doug Tygar

*University of California, Berkeley, CA, USA*

Moshe Y. Vardi

*Rice University, Houston, TX, USA*

Gerhard Weikum

*Max-Planck Institute of Computer Science, Saarbruecken, Germany*

Andreas Butz Brian Fisher  
Antonio Krüger Patrick Olivier (Eds.)

# Smart Graphics

5th International Symposium, SG 2005  
Frauenwörth Cloister, Germany, August 22-24, 2005  
Proceedings



Springer

## Volume Editors

Andreas Butz  
Ludwig-Maximilians-Universität München  
Institut für Informatik, LFE Medieninformatik  
Amalienstrasse 17, 80333 München, Germany  
E-mail: butz@ifi.lmu.de

Brian Fisher  
Simon Fraser University at Surrey  
13450 102 Ave, Surrey BC V3T 5X3, Canada

Antonio Krüger  
Westfälische Wilhelms-Universität  
Institute for Geoinformatics  
Robert-Koch-Str. 26-28, 48149 Münster, Germany  
E-mail: antonio.krueger@gmail.com

Patrick Olivier  
University of Newcastle upon Tyne  
Informatics Research Institute  
Newcastle upon Tyne NE1 7RU, UK  
E-mail: p.l.olivier@ncl.ac.uk

Library of Congress Control Number: 2005930515

CR Subject Classification (1998): I.3, I.2.10, I.2, I.4, I.5, H.5, I.7

ISSN	0302-9743
ISBN-10	3-540-28179-7 Springer Berlin Heidelberg New York
ISBN-13	978-3-540-28179-5 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

[springeronline.com](http://springeronline.com)

© Springer-Verlag Berlin Heidelberg 2005  
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India  
Printed on acid-free paper SPIN: 11536482 06/3142 5 4 3 2 1 0

# Preface

The International Symposium on Smart Graphics 2005 was held from August 22–24, 2005 in the cloister Frauenwörth on the island of Frauenchiemsee, Germany. It was the sixth event in a series which originally started in 2000 as a AAAI Spring Symposium.

In response to the overwhelming success of the 2000 symposium, its organizers decided to turn it into a self-contained event. With the support of IBM, the first two International Symposia on Smart Graphics were held at the T.J. Watson Research Center in Hawthorne, NY in 2001 and 2002. The 2003 symposium moved to the European Media Lab in Heidelberg to underline the international character of the Smart Graphics enterprise and its community.

The core idea behind these symposia is to bring together researchers and practitioners from the field of computer graphics, artificial intelligence, cognitive science, graphic design and the fine arts. Each of these disciplines contributes to what we mean by the term “Smart Graphics”: the intelligent process of creating effective, expressive and esthetic graphical presentation. Many Smart Graphics symposia emphasize a particular aspect of the field in the call for papers. In 2005 our focus was on “Visual Analytics” – the art and science of analytical reasoning facilitated by interactive visual interfaces.

While artists and designers have been creating communicative graphics for centuries, artificial intelligence focuses on automating this process by means of the computer. While computer graphics provides the tools for creating graphical presentations in the first place, the cognitive sciences contribute the rules and models of perception necessary for the design of effective graphics. The exchange of ideas between these four disciplines has led to many exciting and fruitful discussions, and the Smart Graphics symposia draw their liveliness from a spirit of open minds and the willingness to learn from and share with other disciplines.

We would like to thank all authors for the effort that went into their submissions, the program committee for their work in selecting and ordering contributions for the final program, and of course the participants who transformed Frauenchiemsee for three days in August into Smart Graphics Island 2005.

August 2005

Andreas Butz  
Brian Fisher  
Antonio Krüger  
Patrick Olivier



# Organization

## Organization Committee

Andreas Butz (University of Munich, Germany)  
Brian Fisher (University of British Columbia, Canada)  
Antonio Krueger (University of Münster, Germany)  
Patrick Olivier (University of Newcastle Upon Tyne, UK)

## Program Committee

Elisabeth Andre (University of Augsburg)  
Marc Cavazza (Teeside University)  
Marc Christie (University of Nantes)  
Sarah Diamond (Banff Centre)  
Steven Feiner (Columbia University)  
Sid Fels (University of British Columbia)  
Knut Hartmann (University of Magdeburg)  
Rainer Malaka (European Media Lab)  
Karol Myszkowski (Max-Planck-Institute Saarbrücken)  
Shigeru Owada (University of Tokyo)  
W. Bradford Paley (Digital Image Design)  
Bernhard Preim (University of Magdeburg)  
Thomas Rist (University of Applied Sciences, Augsburg)  
Stefan Schlechtweg (University of Magdeburg)  
Thomas Strothotte (University of Magdeburg)  
Lucia Terrenghi (University of Munich)  
Sha Xinwei (Georgia Institute of Technology)  
Massimo Zancanaro (ITC-irst Trento)  
Michelle Zhou (IBM T.J. Watson Research Center)

## Secondary Reviewers

Alexander Belyaev (Max-Planck-Institute Saarbrücken)  
Angela Brennecke (University of Magdeburg)  
Tobias Isenberg (University of Calgary)  
Grzegorz Krawczyk (Max-Planck-Institute Saarbrücken)

## Supporting Institutions

The Smart Graphics Symposium 2005 was held in cooperation with Eurographics, AAAI and ACM Siggraph.

# Table of Contents

## Synthetic Characters and Virtual Worlds

Engaging in a Conversation with Synthetic Characters Along the  
Virtuality Continuum

*Elisabeth André, Klaus Dorfmueller-Ulhaas, Matthias Rehm* . . . . . 1

Visualizing Emotion in Musical Performance Using a Virtual Character

*Robyn Taylor, Pierre Boulanger, Daniel Torres* . . . . . 13

Knowledge in the Loop: Semantics Representation for Multimodal  
Simulative Environments

*Marc Erich Latoschik, Peter Biermann, Ipke Wachsmuth* . . . . . 25

Virtual Camera Planning: A Survey

*Marc Christie, Rumesh Machap, Jean-Marie Normand,  
Patrick Olivier, Jonathan Pickering* . . . . . 40

## Generating Visual Displays

Graphical Data Displays and Database Queries: Helping Users Select  
the Right Display for the Task

*Beate Grawemeyer, Richard Cox* . . . . . 53

Visualization Tree, Multiple Linked Analytical Decisions

*José F. Rodrigues Jr., Agma J.M. Traina, Caetano Traina Jr.* . . . . . 65

Structure Determines Assignment Strategies in Diagrammatic  
Production Scheduling

*Rossano Barone, Peter C.-H. Cheng* . . . . . 77

Generation of Glyphs for Conveying Complex Information, with  
Application to Protein Representations

*Greg D. Pintilie, Brigitte Tuekam, Christopher W.V. Hogue* . . . . . 90

Negotiating Gestalt: Artistic Expression by Coalition Formation  
Between Agents

*Kaye Mason, Jörg Denzinger, Sheelagh Carpendale* . . . . . 103

**Text and Graphics**

Metrics for Functional and Aesthetic Label Layouts  
*Knut Hartmann, Timo Götzelmann, Kamran Ali,  
Thomas Strothotte* ..... 115

A Smart Algorithm for Column Chart Labeling  
*Sebastian Müller, Arno Schödl*..... 127

**3D Interaction and Modeling**

Usability Comparison of Mouse-Based Interaction Techniques for  
Predictable 3d Rotation  
*Ragnar Bade, Felix Ritter, Bernhard Preim* ..... 138

Multi-level Interaction in Parametric Design  
*Robert Aish, Robert Woodbury* ..... 151

Intuitive Shape Modeling by Shading Design  
*Bertrand Kerautret, Xavier Granier, Achille Braquelaire* ..... 163

Automatic Cross-Sectioning Based on Topological Volume  
Skeletonization  
*Yuki Mori, Shigeo Takahashi, Takeo Igarashi, Yuriko Takeshima,  
Issei Fujishiro* ..... 175

**Novel Interaction Paradigms**

Interface Currents: Supporting Fluent Collaboration on Tabletop  
Displays  
*Uta Hinrichs, Sheelagh Carpendale, Stacey D. Scott, Eric Pattison* ... 185

Design of Affordances for Direct Manipulation of Digital Information  
in Ubiquitous Computing Scenarios  
*Lucia Terrenghi* ..... 198

2D Drawing System with Seamless Mode Transition  
*Yoshihito Ohki, Yasushi Yamaguchi* ..... 206

**Poster Presentations and Demos**

Tentative Results in Focus-Based Medical Volume Visualization  
*Timo Ropinski, Frank Steinicke, Klaus Hinrichs* ..... 218

3d Visualisation in Spatial Data Infrastructures <i>Torsten Heinen, Martin May, Benno Schmidt</i> .....	222
From Artefact Representation to Information Visualisation: Genesis of Informative Modelling <i>Iwona Dudek, Jean-Yves Blaise</i> .....	230
Computer-Assisted Artistic Pattern Drawing <i>Hun Im, Jong Weon Lee</i> .....	237
VR-Mirror: A Virtual Reality System for Mental Practice in Post-Stroke Rehabilitation <i>Jose A. Lozano, Javier Montesa, Mari C. Juan, Mariano Alcañiz, Beatriz Rey, Jose Gil, Jose M. Martinez, Andrea Gaggioli, Francesca Morganti</i> .....	241
Picturing Causality – The Serendipitous Semiotics of Causal Graphs <i>Eric Neufeld, Sonje Kristtorn</i> .....	252
Xface: Open Source Toolkit for Creating 3D Faces of an Embodied Conversational Agent <i>Koray Balci</i> .....	263
Interactive Augmentation of 3D Statistics Visualizations <i>Peter Bergdahl</i> .....	267
<b>Author Index</b> .....	269

# Engaging in a Conversation with Synthetic Characters Along the Virtuality Continuum

Elisabeth André, Klaus Dorfmueller-Ulhaas, and Matthias Rehm

Multimedia Concepts and Applications, University of Augsburg, Germany  
andre@informatik.uni-augsburg.de

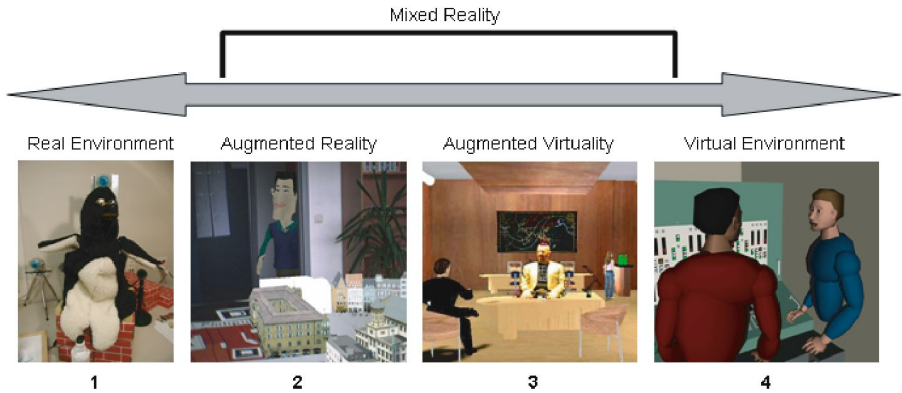
**Abstract.** During the last decade research groups as well as a number of commercial software developers have started to deploy embodied conversational characters in the user interface especially in those application areas where a close emulation of multimodal human-human communication is needed. Most of these characters have one thing in common: In order to enter the user's physical world, they need to be physical themselves. The paper focuses on challenges that arise when embedding synthetic conversational agents in the user's physical world. We will start from work on synthetic agents that populate virtual worlds and anthropomorphic robots that inhabit physical worlds and discuss how the two areas need to be combined in order to populate physical worlds with synthetic characters. Finally, we will report on so-called traversable interfaces that allow agents to cross the border from the physical space to the virtual space and vice versa.

## 1 Introduction

The objective to develop more human-centered, personalized and at the same time more entertaining interfaces immediately leads to the metaphor of an embodied conversational agent that employs gestures, mimics and speech to communicate with the human user. Incarnations of such characters differ widely in type and amount of embodiment – starting from simplistic cartoon-style 2D representations of faces, fully embodied virtual humans in 3D virtual worlds to physically embodied androids co-habiting the user's real world (see [1] for an overview). Despite of their variety, most of these characters have one thing in common: In order to get in touch with a user in the physical space, the characters have to be physical themselves.

Following [2], we may classify the contact between synthetic and human agents according to a "virtuality continuum" (see Fig. 1). At one extreme, we find android agents that are completely integrated in the user's physical world and even allow for physical contact with the user. Mel, a robotic penguin developed by Sidner and colleagues [3] (see image 1 in Fig. 1), is one of the most sophisticated physical agents that engages in face-to-face communication with a human user. At the other extreme, there are purely virtual environments that are populated by human and synthetic agents. A prominent example is the pedagogical agent Steve [4] (see Image 4 in Fig. 1). Steve is aware of the user's presence in the virtual space, monitors her actions and responds to them, but has no access to the external world. That is it is only able to perceive user actions that are performed in the virtual space. In between, we find a new generation of characters that inhabit a world in which virtual and digital objects are smoothly integrated. In these

applications, projections of virtual characters overlay the user's physical environment or projections of real persons are inserted into a virtual world. For instance, Cavazza and colleagues [5] propose a magic mirror paradigm which puts the user both in the role of an actor and a spectator by inserting the user's video image in a virtual world that is populated by synthetic agents (see Image 3 in Fig. 1). Kruppa and colleagues [6] focus on the reciprocal problem, namely how to populate the user's physical environment with synthetic agents. Their character is capable of freely moving along the walls of a real room by making use of a steerable projector and a spatial audio system. The Virtual Augsburg project aims at an even tighter integration of a character into a physical world (see [7]). In this application, a synthetic character called Ritchie jointly explores with the user a table-top application that combines virtual buildings of the city center of Augsburg with a real city map being laid out on a real table. Augmented Reality applications are characterized by a tight spatial connection between physical and virtual objects. This includes the correct projection of occlusions as well as the generation of realistic shadows falling onto real objects. The second image in Fig. 1 shows the character Ritchie when entering the Multimedia Lab. As shown in the image, the real door frame partially covers, for instance, the virtual character which in turn occludes the wall in the background.



**Fig. 1.** Milgram's Diagram of Virtuality Continuum Adapted to Embodied Conversational Characters: the robotic penguin developed by Sidner and colleagues [3] at MERL (image 1), the virtual character Ritchie entering the Multimedia Interfaces Lab at Augsburg University (image 2), Marc Cavazza acting as "Goldfinger" in an Augmented Virtuality application developed by his team at Teaside University [5] (image 3), the pedagogical agent Steve developed by Rickel and Johnson at ISI [4] (image 4)

Most work so far has concentrated on the design and implementation of conversational agents at the two extremes of the Virtuality Continuum. The objective of this paper is to investigate which challenges arise from embedding synthetic conversational characters in the user's physical world. We will start from work on synthetic agents populating virtual worlds as well as anthropomorphic robots inhabiting physical worlds and discuss how the two areas need to be combined in order to enable characters to

enter worlds between the two extremes of the Virtuality Continuum and engage in a conversation with a human user. In particular, we are interested in grounding, i.e. how to establish a common understanding between a user and a synthetic agent of what is being said and meant at the various levels of the Virtuality Continuum. Furthermore, we will describe challenges resulting from so-called traversable interfaces which allow a character to cross the border from the real to the virtual world and vice versa.

## 2 Perceptual Attention

Face-to-face communication does not take place in an empty space, but should be linked to the surroundings of the conversational partners. An embodied agent may acquire knowledge about its surroundings by a sensory mechanism and/or by consulting a model of the interaction space that might be mentioned during the interaction. In general we can distinguish between synthetic vision and real vision approaches. The first refers to models of attention and perception in the virtual world whereas the latter comprises efforts to recognize and interpret nonverbal input from the real world, such as objects in the interaction space or a user's gestures and head movements.

Synthetic vision approaches model (part of) the human perceptual process in an agent which is situated in a virtual environment. The agent is supplied with (virtual) visual sensors which are used to extract visual information about the environment. A characteristic feature of all approaches in this area is the idea to enable only limited access to information about the world. On the one hand, this allows for a more natural attention behavior of the agent because it is no longer omniscient but has to actively look around to gather necessary information. On the other hand, it is important to reduce the processing load of the agent. If the world becomes very large, the price to pay for omniscience is the need for much computational power. Another feature is the distinction between a bottom-up and a top-down control of attention. Bottom-up control is involuntary and triggered e.g. by the saliency of an object (bright color, movement) whereas top-down control is a goal-directed volitional act, e.g. finding a grey ball which will focus attention on round objects with less bright colors.

Peters and O'Sullivan [8] describe a model for bottom-up visual attention that attempts to mimic the low-level, automatic mechanisms of early visual processing which rely on salience features and draw attention to salient locations in the input. Thus, their agent is capable of idle or spontaneous looking which takes place when there is no task to perform. It can also be employed in case of a necessary task interruption, e.g. in case of a threat, such as a moving car. Their model combines database queries (on the velocity of objects) with a visual sensor. Due to the restricted access to the scene database, the model is enhanced with an object-based short-term memory. Input to the memory component are simplified images without textures and lighting that simulate peripheral and foveal vision. In the case of peripheral vision only groups of objects are recognized.

Hill [9] proposes a two step model of perceptual attention for virtual agents that focuses mainly on the problem of object grouping to prevent a conceptual overload. By grouping similar objects, they can be treated as one object. He distinguishes between

an automatic process of grouping which he describes as pre-attentive and a process of attention control. Automatic grouping segments the input and groups objects according to Gestalt principles, such as proximity or similarity. Groups are created, joined or split making use of the center of mass defined by similar objects and a fixed grouping radius that originates in this center of mass. Attention control then is a voluntary process that can be triggered in two ways: (i) bottom-up by salient external stimuli, such as luminance, color or motion, (ii) top-down by the cognitive system in which case attention is task-oriented, e.g., for searching or tracking objects.

Chopra and Badler [10] present a psychologically-motivated framework for automating a character's visual attention. It implements a model of eye behavior where repetitive cycles of eye movements constitute pre-defined patterns of looking behaviors that are associated with motor activities (walk, reach, etc.) as well as with cognitive actions (search, track, etc.). The authors emphasize the importance of the agent's task for its attentive behaviors. Without a task, the agent exercises spontaneous looking where attention is easily captured by peripheral events. In case the agent engages in a task, attention is more focused rejecting such an exogenous capturing of attention.

Whereas synthetic vision approaches try to model human perceptual and attentive processes in the virtual environment, computer vision approaches are rather concerned with capturing actions and events in the real world. In contrast to synthetic vision, the real world is much less predictable, more complex and dynamic as the virtual world. Thus, the goal cannot be a simulation of the human perceptual process, but the recognition of certain real world features to further a more natural interaction. Examples of such recognition tasks are head tracking which allows the agent to maintain mutual gaze or interpret the user's looking behaviors, gesture recognition and interpretation for identifying points of interest the user indicates, or recognition of objects that can be used as shared references.

The virtual agent Max (e.g. see [11] or [12]) employs a number of different sensors to collect information from the user aiming at the detection of context-sensitive foci. The face recognition system relies on detecting skin colored objects in the upper regions of a camera image allowing the agent to focus on these salient regions and giving the user the impression that the agent is maintaining mutual gaze with the user. Capturing the user's gestures just from a visual camera image is error-prone, resulting in the use of data gloves and a corresponding tracking system.

Agents that inhabit Augmented Realities need to be aware of the physical as well as the digital space. As a consequence, we have to fuse the results from the synthetic and real vision processes. One specific problem is the different information density for the real and for the virtual world. Perception and attention of the virtual part of the scenario can be modelled to whatever degree is desirable because all the information about the virtual world is in principle available. Perceiving and attending to the real world faces severe technical problems and is thus always limited and deficient unless the surrounding real world is static and can thus be modelled as well. Furthermore, new methods are required for calculating attentional prominence in an Augmented Reality which take into account both physical and digital object features.



### 3 Engagement Cues in Conversation

It does not make much sense to implement mechanisms for perceptive behaviors if the agent is not able to show its awareness for the environment. In this paper, we focus on attention in a conversation as an important means to show engagement in a dialogue. Sidner and colleagues [3] refer to engagement as “the process by which two (or more) participants establish, maintain and end their perceived connection during interactions they jointly undertake”. Engagement behaviors may be directed to the conversational partner as well as the conversational environment. Clark [13] consider attention to the speaker as an important signal of understanding by the addressee. Furthermore, Nakano and colleagues [14] observed that shared attention to an object may be interpreted as a sign of common understanding for the task at hand.

#### 3.1 Engagement Cues to the Conversational Environment

To indicate attention to objects in the environment, Clark [15] proposes several directing-to and placing-for behaviors.

Directing-to behaviors may be realized by a number of verbal and non-verbal means including demonstrative pronouns, eye gaze or pointing gestures. In an Augmented Reality, directing-to behaviors may relate to both virtual and physical objects. For instance, the character Ritchie in Image 2 in Fig. 1 might generate a mixed deictic utterance, such as “the building in the middle of the table” to help the user to localize a digital object in the physical space. Existing algorithms for the analysis and generation of deictic referring expressions usually start from the assumption that all reference objects are located in the same space - either virtual (see [16]) or physical (e.g. see [17]). To extend these algorithms to the broader context of mixed realities, real and digital objects have to be localized within a uniform coordinate system. Furthermore, we need to take into account the real and digital properties of objects as well the position of real and virtual conversational partners. For instance, to generate a deictic utterance, such as “the building on the table in front of you” in Image 2 in Fig. 1, the system needs to know the relative position of a physical object (the table), a digital object (the building) and the user in the real space.

Placing-for behaviors differ from directing-to behaviors by moving objects in the addressee’s focus of attention. For instance, a customer may place a product on the counter to signal the cashier that she wishes to buy the product. Interestingly, such behaviors serve to provide common ground between the customer and the cashier regarding the action to be performed. That is they are usually understood without any additional verbal comment, just by considering the current context. Augmented Reality interfaces offer novel forms of placing-for behaviors that are situated in a physical space. For instance, a synthetic character may place a virtual object on a real table to direct the user’s attention to this object. Vice versa, a human user may position physical objects in the character’s focus of attention. In the Virtual Augsburg application, the user may position a cardboard box in the physical scene to signal the character where it should move (see Fig. 2). A frustum of pyramid with markers attached to each side is used to facilitate the recognition of the position and orientation of the cardbox employing the user’s head-worn camera.



**Fig. 2.** Placing a Physical Object in the Real Space to Attract a Character’s Attention

### 3.2 Engagement Cues to the Conversational Partner

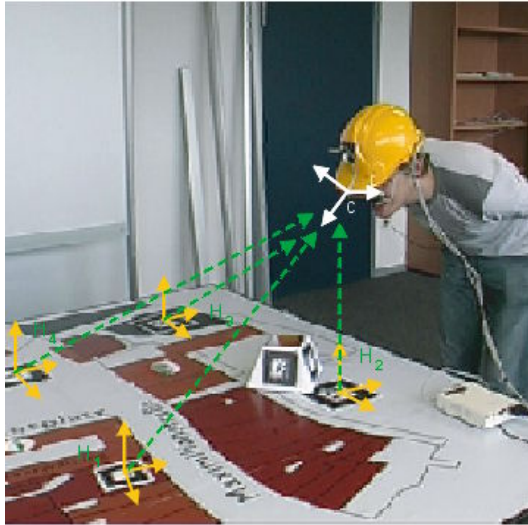
A number of researchers focus on eye gaze as one of the most important means to indicate engagement in a dialogue. According to Kendon [18], we can distinguish between at least four functions of seeking or avoiding to look at the partner in dyadic interactions: (i) to provide visual feedback, (ii) to regulate the flow of conversation, (iii) to communicate emotions and relationships, (iv) to improve concentration by restriction of visual input. Concerning the listener, Argyle and Cook [19] show that people look nearly twice as much while listening than while speaking. Compared to dyadic conversations, we know little about gaze behavior in multiparty interaction. Vertegaal and colleagues [20] describe a study of the looking behavior in a four-party interaction. Subjects looked about 7 times more at the individual they listened to than at others. They looked about three times more at the individual they spoke to than at others. In accordance with Sidner et al. [3] or Nakano et al. [14], they conclude that gaze, or looking at faces, is an excellent predictor of conversational attention in multiparty conversations.

Empirical studies of engagement in human-human conversation have inspired the implementation of engagement behaviors for embodied conversational agents. Nakano and colleagues [14] developed an engagement model for the kiosk agent Mack that provides route directions for a paper map. The agent uses gaze as a deictic device as well as a feedback and turn taking mechanism. Based on an analysis of human-human conversation, Sidner and colleagues [3] implemented a conversational robot that is able to track the face of the conversational partner and adjusts its gaze towards him or her. Even though the set of communicative gestures of the robot was strongly limited, an empirical study revealed that users indeed seem to be sensitive to a robot’s conversational gestures and establish mutual gaze with it. Nakano and Nishida [21] present a synthetic agent which recognizes the user’s gaze direction as a cue to non-verbal engagement. The agent is situated on a large screen in front of a static background image which offers a number of points of interest that can be brought up during the conversation. The

conversation is influenced by the conversational and the perceptual situation, i.e., the user's focus of attention given by his/her gaze direction.

In order to enable an agent to reason about another agent's attentive behaviors, Peters [22] proposes to rely on a theory of mind. The model by Baron-Cohen [23] seems to offer a promising approach since it relates gaze perception to higher level cognitive processes. Based on this model, Peters proposes the implementation of a synthetic vision module that considers to what extent the orientation of an agent's eyes, head and body are directed towards the conversational partner in order to determine its attention towards the conversational partner.

In the Virtual Augsburg Scenario, the user's head position and orientation is tracked to monitor his or her focus of attention. The tracking is enabled by a small camera fixed on the user's head-mounted display (see Fig. 3) and a marker tracking software running on a standard PC.<sup>1</sup> In this way, the character may intervene if the user is losing interest. Nevertheless, the character is only able to reason about the user's attention towards the digital world and a strongly limited section of the real world (essentially the table) so far.



**Fig. 3.** Tracking System for Virtual Augsburg

## 4 Conversational Locomotion

While most work on face-to-face communication focuses on the generation of gestures, mimics and speech, hardly any research has addressed the problem of conversational locomotion. Nevertheless, face-to-face communication often requires the participants to change their position in space. In a party-like scenario, people dynamically join or

<sup>1</sup> We have started with the wide-spread AR Toolkit. However, we have created our own tracking system which relies on improved marker detection and pose estimation processes.

leave groups with which they engage in a communication. Furthermore, people often move to another location in order to be closer to objects they wish to talk about.

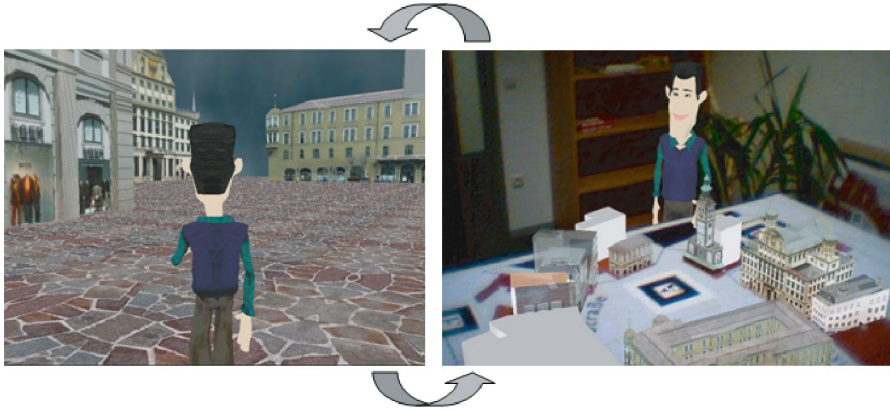
An early approach to coordinate locomotive, gestural, and speech behaviors for an embodied conversational agent has been presented by Lester and colleagues [24]. Peters [22] discusses the use of locomotive behaviors, directed gestures, body orientation and gaze to express different levels of attention to objects and persons. Thalmann and colleagues [25] concentrate on the simulation of social navigation behaviors in virtual 3D environments including the social avoidance of collisions, intelligent approach behaviors, and the calculation of suitable interaction distances and angles. The work is based on an operationalization of empirically-grounded theories of human group dynamics, such as Kendon's group formation system [18]. To enable such a behavior in an Augmented Reality, the agent needs to navigate through a physical space, thereby avoiding collisions with physical and digital objects.

The question arises, however, of whether humans follow similar social group dynamics when interacting with synthetic agents in Virtual or Augmented Realities. To explore social navigation behaviors of humans in immersive 3D worlds, Bailenson and colleagues [26] have conducted a number of empirical studies which revealed that human subjects apply similar behavior patterns in a virtual world as in a physical world. For instance, the size and the shape of the personal space around the virtual humans resembled the shape of the space that people leave around real, nonintimate humans. Another interesting result was the observation that the social navigation behavior depends on whether the agent is controlled by a computer or by a real human. They found out that people gave an avatar more personal space than an autonomous agent. In a second experiment, they examined what happens when virtual humans walk and violate the personal space of the participants. The experiment showed that human subjects avoided the virtual human more when it was obviously computer-controlled than an avatar that was controlled by another human.

In the experiments above, the user was completely immersed in the real world. Naturally, the question arises of whether the same findings hold for social navigation behaviors between human and virtual agents in an Augmented Reality. Most likely the navigation behavior will depend on how smoothly the character is integrated in the physical world. In particular, deficiencies of the employed display technology might disturb the user's sense of sharing a space with the character and lead to unexpected or little natural behaviors.

## 5 Traversable Interfaces

The preceding sections started from scenarios in which characters were bound to a specific space: a real, virtual or augmented environment. An intriguing challenge is the realization of so-called traversable interfaces [27] that allow human and synthetic agents to cross the border from the digital world to the real world and vice versa. One advantage of traversable interfaces lies in the fact that it allows users to move to another interaction space if the information presentation requires it. An example of a traversable interface is shown in Fig. 4. The character Ritchie is first exploring with the user a virtual 3D representation of the city of Augsburg. It then moves to the real space to study a miniaturized model of the city of Augsburg laid out on a table.



**Fig. 4.** Moving from the Real to the Virtual World and Vice Versa

In research on virtual reality, a higher sense of presence has been considered as a critical success for many applications, such as virtual meeting rooms or training applications. Witmer and Singer [28] define *presence* as “the subjective experience of being in one place or environment, even when one is physically situated in another”. Usually, it will take the users some time to get a sense of presence for a new world. Traversable interfaces need to provide for means that enable the user a smooth transfer to another world. It may help the user if the system announces the transfer to another interaction space, for example, by employing beam techniques as in *Star Trek* (see [29]). In order to help the users to orient themselves, we make use of a virtual companion as a uniform interaction metaphor that accompanies them not only in the digital, but also in the real space. For instance, on the left-hand side of Fig. 4, user and character explore a virtual 3D model of Augsburg from a pedestrian’s angle. In our case, there is no visual representative of the user in the virtual 3D space which is supposed to follow the agent looking over its shoulder.<sup>2</sup> On the right-hand side of Fig. 4, user and character are both situated in the physical world and study the city model of Augsburg from a bird’s eye view. In both cases, the user perceives the scene in a similar way as the character does in order to make it easier for her to share experiences with it.

In order to establish a temporally persistent relationship with the user, the agent should maintain a memory of joint experiences in the different worlds. For instance, when visiting 3D model of the town hall, the agent should remember what it told the user about the corresponding table-top model. To allow for references from one world to the other, we need to maintain a dialogue history across interaction spaces. To our knowledge, no research so far has investigated how mechanisms for the generation and analysis of anaphora need to be adapted to cope with traversable interfaces. So-called alignment problems may arise if the agent produces referring expressions for objects in

<sup>2</sup> We still need to investigate whether the missing embodiment of the user in the 3D space leads to a too drastical lost of presence. In this case, it might help to visualize parts of the user’s body, such as her forearms.

the new interaction space which the user is not able to resolve because she has not yet succeeding in developing a sense of presence for the new world.

We are currently preparing an empirical study to investigate in how far the use of a virtual companion may help the user to move from one space to the other. In particular, we are interested in the question of whether the use of a virtual character contributes to a more coherent perception of the interaction.

## 6 Conclusion

In this paper, we have presented a new generation of synthetic characters that are no longer bound to a flat screen, but able to enter a physical world and to engage in a conversation with a human user. Users and characters do not inhabit separated spaces, but share an informational and physical reality that is augmented by digital objects. As a consequence, communication has to take into account both the physical and the digital context. New forms of deixis are enabled by the manipulation of objects and movements of characters in the physical space. Further challenges arise from the realization of so-called traversable interfaces that allow human and synthetic agents to cross the border from the digital world to the real world and vice versa. Obviously, mixed realities put high demands to the agents' perceptive skills requiring to integrate mechanisms for real and synthetic vision. It remains a great challenge to generate plausible engagement behaviors despite of seriously impoverished perception channels.

## Acknowledgement

This paper was partially funded by the EU Network of Excellence Humaine. We would like to thank Peter Rist for the graphical design of the character Ritchie. We are also grateful to Lewis Johnson, Fred Charles and Candy Sidner for giving us the permission to use their figures. The copyright for the figures is to the single institutions.

## References

1. Gratch, J., Rickel, J., André, E., Badler, N., Cassell, J., Petajan, E.: Creating interactive virtual humans: Some assembly required. *IEEE Intelligent Systems* **17** (2002) 54–63
2. Milgram, P., Kishino, F.: A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems* **E77-D** (1994) 1321–1329
3. Sidner, C.L., Kidd, C.D., Lee, C., Lesh, N.: Where to look: a study of human-robot engagement. In: *IUI '04: Proceedings of the 9th international conference on Intelligent user interface*, New York, NY, USA, ACM Press (2004) 78–84
4. Jeff Rickel, W.L.J.: Animated agents for procedural training in virtual reality: Perception, cognition, and motor control. *Applied Artificial Intelligence* **29** (1999) 343–382
5. Cavazza, M., Charles, F., Mead, S.J., Martin, O., Marichal, X., Nandi, A.: Multimodal acting in mixed reality interactive storytelling. *IEEE-Multimedia* **11** (2004) 30–39
6. Kruppa, M., Spassova, L., Schmitz, M.: The virtual room inhabitant. In: *Proc. of the IUI-2005 Workshop on Multi-User and Ubiquitous User Interfaces (MU3I)*, San Diego, USA (2005)



7. André, E., Dorfmueller-Ulhaas, K., Rist, T.: Embodied conversational characters: Wandering between the digital and the physical world. *it - Information Technology* **46** (2004) 332–340
8. Peters, C., O'Sullivan, C.: Bottom-up Visual Attention for Virtual Human Animation. In: *Proceedings of the Computer Animation and Social Agents (CASA) 2003*, Rutgers University, New York. (2003) 111–117
9. Hill, R.: Modeling Perceptual Attention in Virtual Humans. In: *Proceedings of the 8th Conference on Computer Generated Forces and Behavioral Representation*, Orlando, FL. (1999)
10. Chopra-Khullar, S., Badler, N.I.: Where to look? Automating attending behaviors of virtual human characters. In: *Proceedings of the third annual conference on Autonomous Agents*, New York, ACM Press (1999) 16–23
11. Jung, B., Kopp, S.: Flurmax: An interactive virtual agent for entertaining visitors in a hallway. In et al., T.R., ed.: *Intelligent Virtual Agents*. Springer, Berlin, Heidelberg (2003) 23–26
12. Kopp, S., Jung, B., Lessmann, N., Wachsmuth, I.: Max — A Multimodal Assistant in Virtual Reality Construction. *KI – Künstliche Intelligenz* (2003) 11–17
13. Clark, H.: *Using Language*. Cambridge University Press, Cambridge (1996)
14. Nakano, Y., Reinstein, G., Stocky, T., Cassell, J.: Towards a model of face-to-face grounding. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2003)*. (2003) 553–561
15. Clark, H.: Pointing and Placing. In Kita, S., ed.: *Pointing: Where language, culture, and cognition meet*. Erlbaum, Hillsdale, N.J. (2003) 1–17
16. Gorniak, P., Roy, D.: A visually grounded natural language interface for reference to spatial scenes. In: *ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces*, New York, NY, USA, ACM Press (2003) 219–226
17. Roy, D., Hsiao, K.Y., Mavridis, N.: Mental imagery for a conversational robot. *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS* **34** (2004) 1374–1383
18. Kendon, A.: *Nonverbal Communication, Interaction, and Gesture*. Mouton, The Hague (1981)
19. Argyle, M., Cook, M.: *Gaze and Mutual Gaze*. Cambridge University Press, Cambridge (1976)
20. Vertegaal, R., Slagter, R., van der Veer, G., Nijholt, A.: Eye gaze patterns in conversations: there is more to conversational agents than meets the eyes. In: *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, New York, NY, USA, ACM Press (2001) 301–308
21. Nakano, Y.I., Nishida, T.: Awareness of Perceived World and Conversational Engagement by Conversational Agents. In: *Proceedings of the AISB 2005 Symposium on Conversational Informatics for Supporting Social Intelligence & Interaction*. (2005)
22. Peters, C.: Foundations of an Agent Theory of Mind Model for Conversation Initiation in Virtual Environments. In: *Proceedings of the AISB 2005 Joint Symposium on Virtual Social AGents*. (2005) 163–170
23. Baron-Cohen, S.: How to build a baby that can read minds: Cognitive Mechanisms in Mind-Reading. *Cahiers de Psychologie Cognitive* **13** (1994) 513–552
24. Lester, J., Voerman, J., Towns, S., Callaway, C.: Deictic believability: Coordinating gesture, locomotion, and speech in lifelike pedagogical agents. *Applied Artificial Intelligence* **29** (1999) 383–414
25. Guye-Vuillième, A., Thalmann, D.: A high level architecture for believable social agents. *Virtual Reality Journal* **5** (2001) 95–106
26. Bailenson, J.N., Blasovich, J., Beall, A.C., Loomis, J.M.: Interpersonal distance in immersive virtual environments. *Personality and Social Psychology Bulletin* **29** (2003) 819–833

27. Koleva, B., Schnädelbach, H., Benford, S., Greenhalgh, C.: Traversable interfaces between real and virtual worlds. In: CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems, New York, NY, USA, ACM Press (2000) 233–240
28. Witmer, B., Singer, M.: Measuring presence in virtual environments: A presence questionnaire. *Presence: Teleoperators and Virtual Environments* **7** (1998) 225–240
29. Rocchi, C., Stock, O., Zancanaro, M., Kruppa, M., Krüger, A.: The museum visit: generating seamless personalized presentations on multiple devices. In: IUI '04: Proceedings of the 9th international conference on Intelligent user interface, New York, NY, USA, ACM Press (2004) 316–318



# Visualizing Emotion in Musical Performance Using a Virtual Character

Robyn Taylor<sup>1</sup>, Pierre Boulanger<sup>2</sup>, and Daniel Torres<sup>3</sup>

<sup>1</sup> Advanced Man-Machine Interface Laboratory,  
Department of Computing Science, University of Alberta,  
T6G 2E8 Edmonton, Alberta, Canada

robyn@cs.ualberta.ca

<sup>2</sup> pierreb@cs.ualberta.ca

<sup>3</sup> dtorres@cs.ualberta.ca

**Abstract.** We describe an immersive music visualization application which enables interaction between a live musician and a responsive virtual character. The character reacts to live performance in such a way that it appears to be experiencing an emotional response to the music it ‘hears.’ We modify an existing tonal music encoding strategy in order to define how the character perceives and organizes musical information. We reference existing research correlating musical structures and composers’ emotional intention in order to simulate cognitive processes capable of inferring emotional meaning from music. The ANIMUS framework is used to define a synthetic character who visualizes its perception and cognition of musical input by exhibiting responsive behaviour expressed through animation.

## 1 Introduction

In his influential 1959 work, “The Language of Music,” Deryck Cooke analyses Western tonal music in an attempt to understand how a musical piece conveys emotional content to a listening audience [2].

Cooke uses the term ‘content’ to represent the sense of exhilaration, despair, or bliss which a listener might report feeling after listening to a powerful piece of music. Although content cannot easily be extracted from the written score in the same way that pitch names, key signatures or harmonic structure may be, Cooke defends his belief that emotional content is inherent to music. He describes content as “not a technical part [of a piece of music], but something more elusive: the interpretation which we put upon the interaction of the technical elements.”<sup>1</sup>

“In other words, the ‘content’ is inseparable from the music, except as an emotional experience derived from listening to the music. If we use the word to signify ‘the emotion contained in the music’, we must keep clear in our minds that the emotion is contained, not as a necklace in a box,

---

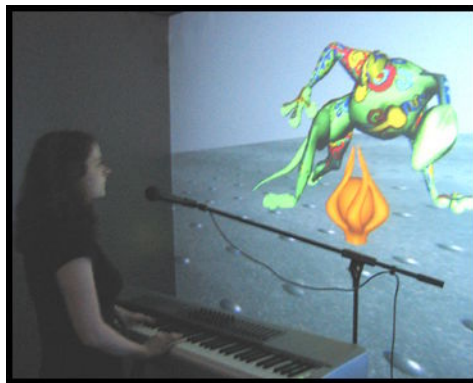
<sup>1</sup> Cooke, *The Language of Music*, p.199.

to be taken out and examined, but as an electric current in the wire: if we touch the wire we shall get a shock, but there is no way whatsoever of making contact with the current without making contact with the wire.”<sup>2</sup>

Cooke’s work references his extensive research in Western tonal music, makes hypotheses concerning the relationships between musical structures and composers’ emotional intent, and cites numerous examples from musical literature to support his conjectures.

We have chosen to rely on Cooke’s findings to implement an immersive music visualization system which attempts to enrich the experience of observing a live musical performance by echoing the musical work’s emotional content – Cooke’s “electric current in the wire” – through the behaviours of a life-sized virtual character who simulates an emotional response to the music being performed.

This paper presents a method of visualizing live musical performance through the behavioural responses of a virtual character. Torres and Boulanger’s ANIMUS Project [1] enables the creation of animated characters that respond in real-time to the stimuli they encounter in the virtual world [14] [15]. We describe an ANIMUS character with the ability to perceive musical input and to encode it in a way consistent with previously defined organizational models for tonal music. The character is equipped with a cognition mechanism which relates perceived music-theoretical features to emotional states consistent with Cooke’s findings, simulating an emotional understanding of the music it ‘hears’. Using keyframe animation techniques, the ANIMUS character may express its simulated emotional response through visual animations generated in real-time and displayed on a life-sized stereoscopic screen (see Figure 1). Its expressive behavioural response animations provide a visual accompaniment to the performer’s musical performance.



**Fig. 1.** The User Interacting with the Virtual Character

<sup>2</sup> Cooke, *The Language of Music*, p.199.

In Section 2 of this paper we discuss previous work in the area of immersive music visualization. Section 3 introduces the ANIMUS Project, the framework upon which our virtual characters are based. Section 4 describes our proposed model for organizing extracted musical feature data using methods derived from an existing music theoretical model. Section 5 discusses Cooke’s research into the relationship between music and emotion, and how it can be used to create believable character response behaviour. In Section 6, we describe how character emotion is expressed through interpolated keyframe animation.

## 2 Previous Work

There exist numerous examples of previous research in musical visualization which correlate audio content and associated imagery. Of particular interest to us are real-time systems which leverage high-end visualization technology to create life-sized, compelling imagery. Imagery of this scope blurs the boundary between the physicality of the performer and the virtual world within which he or she is immersed.

“The Singing Tree” [8], created by Oliver *et al.* at MIT’s Media Laboratory, immerses a user inside an artistic space comprised of computer graphics and installed set pieces. The environment is visibly and audibly responsive to the sound of his or her voice. “The Singing Tree” provides audio-visual feedback to participants in order to prompt them towards the goal of holding a prolonged note at a steady pitch.

Jack Ox’s visualizations within a three-walled immersive CAVE system explore the harmonic structure of musical input [9]. Her “Color Organ” allows viewers to visualize harmonic relationships in a musical piece by creating three-dimensional structures and landscapes that observers can explore and navigate at will.

An audio-visual performance installation piece, “Messa di Voce”, created by Golan Levin and Zachary Lieberman [7], relates the physicality of the performers to the physicality of the virtual space within which they are performing. It visualizes abstract representations of live vocalizations which originate from the locations of the vocalists’ mouths. The scale of the imagery makes the performers appear to be a part of the virtual space within which they are performing.

Taylor, Torres, and Boulanger [12] have previously described a system that parameterizes live musical performance in order to extract musical features and trigger simple behaviours in Torres and Boulanger’s ANIMUS characters [14][15]. ANIMUS characters can be displayed upon a large stereoscopic projection screen, enabling performers and audience members to perceive them as life-sized and three-dimensional.

We wish to increase the complexity of our music visualization system by creating examples of ANIMUS characters which both perceive live musical input in a ‘human-like’ fashion and appear to interpret it in such a way as is consistent with Cooke’s research into the relationship between musical features and emo-

tional content. It is our hope that this will provide a novel way of visualizing music through human interaction with responsive virtual characters.

### 3 The ANIMUS Architecture

Torres and Boulanger’s ANIMUS Project [1] is a framework which facilitates the creation of ‘believable’ virtual characters. They describe a believable character as “[appearing] to be alive, giving the illusion of having its own thoughts, emotions, intention and personality” [14].

ANIMUS characters are animated characters which respond to events in their environment in ways that illustrate their particular personalities. ‘Shy’ characters may cringe as if startled when another character makes a sudden movement, while ‘curious’ characters may come forward to investigate changes in their environment.

In order to create these types of responsive characters, Torres and Boulanger break the task of information organization and processing into three layers:

- **Perception Layer:** ANIMUS characters must perceive features and events in the world around them. Examples of perceivable events could include user input or actions of other characters in the virtual world. For our purposes, ANIMUS characters must be able to perceive important features in live musical performance. We will describe, in Section 4, how our specialized Musical Perception Filter Layer makes this possible.
- **Cognition Layer:** The characters must analyze the input they have perceived and determine appropriate response behaviours. In this layer, character ‘personality’ is created by defining how perceived events affect the character’s internal state. In Section 5, we discuss how our ANIMUS characters simulate an emotional understanding of perceived musical features.
- **Expression Layer:** When an ANIMUS character has processed perceived data and determines that physical response behaviour is warranted, these behaviours are expressed through animation. The ANIMUS system generates these animations at run-time by using key-frame animation to interpolate between various combinations of pre-defined poses. Animated behaviours vary in mood and intensity in order to illustrate the virtual character’s cognitive state.

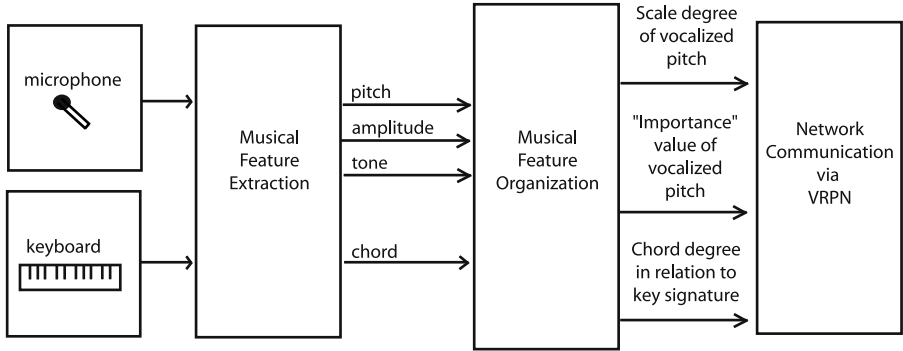
### 4 Identifying Musical Features Through a Specialized Perception Layer

ANIMUS characters receive and share information about the world around them using a ‘blackboard’ system. Information about events occurring within the virtual world is entered on the blackboard, and characters monitor this blackboard in order to perceive these events.

Our ANIMUS characters must be able to ‘listen’ to a live musical performance and perceive meaningful data within that performance. This is necessary in

order for them to assign cognitive meaning to aspects of the music they have ‘heard.’ The stream of incoming live music must be parsed and organized in a perceptually relevant way.

For this purpose, we have created a specialized perception layer called the Musical Perception Filter Layer (see Figure 2). The live musician interfaces with this layer by singing into a microphone and playing a digital piano. The Musical Perception Filter Layer is implemented in a distributed fashion, leveraging the capabilities of a dedicated machine to handle the audio analysis tasks in a real-time manner.



**Fig. 2.** Musical Perception Filter Layer

Our tasks within the Musical Perception Filter Layer are twofold. First we must parse a complex stream of incoming musical data in order to extract meaningful features upon which further analysis may be made. Second, in order to undertake the task of simulating human-like emotional responses to musical input, we must organize these features in a way that is consistent with previous research in the area of human musical perception.

#### 4.1 Extracting Musical Feature Data from Real-Time Audio and MIDI Signals

We use a Macintosh G5 system to extract important features from a stream of live musical input, and then communicate these extracted features across a network to the PC running the ANIMUS engine.

In order to parse the stream of live music, our system uses functionality provided by Cycling '74's Max/MSP [3] development environment. Max/MSP provides users with a graphical environment within which they may create audio applications. There exists a large community of Max/MSP users. Often, developers freely share their ‘patches’ and custom made ‘objects’ with other community members.

We have created a special Max patch combining existing user-created Max objects with our own custom made objects in order to handle the task of extracting features from live musical input.

The features we are interested in extracting are the pitch and amplitude of sung vocals, data describing the singer’s vocal tone quality, and chord information obtained when the user plays upon the digital keyboard.

Feature extraction from sung vocal input is done using the `fiddle~` object created by Puckette *et al.*[11]. The `fiddle~` object extracts information about the singer’s pitch and amplitude. Additionally, `fiddle~` produces raw peak data describing the harmonic spectra of the user’s singing voice.

Upon examination of this harmonic spectra, we define a numerical descriptor indicating whether the user’s vocal tone amplitude is mainly concentrated at the fundamental frequency, or whether there is also significant tone amplitude distributed amongst higher partials. This allows us to numerically describe an aspect of the singer’s vocal timbre. This analysis could be further expanded in the future in order to produce a more detailed measure of vocal timbre, but currently produces a simple parameter that a vocalist can control by modifying the vocal tone he or she employs.

Our own sub-patch monitors MIDI events in order to determine what chords are being played on the keyboard.

## 4.2 A Perceptual Model for Musical Feature Organization

Western tonal music, while at the lowest level consisting of pitches, durations and amplitudes, is constrained by rules of harmonic structure. Experiments in psychoacoustics by Koelsch *et al.* [6] have shown that perceivable deviations from these harmonic structural rules produce noticeable event-related brain potentials (ERPs). These occur even in the brains of non-musicians, indicating that humans exposed to Western tonal music internalize at some level its harmonic structure. Patel *et al.*[10] report that a trained musician’s P600 ERP component responds to harmonic incongruity in the same way as it responds to linguistic incongruity, suggesting similarities between linguistic and harmonic syntactic processing.

Since we are intending to use Western tonal music as input to this system, we have chosen to implement our ANIMUS character’s musical perception skills in such a way as is consistent with tonal music theory. We give our ANIMUS character the ability to identify the harmonic context of the vocalized pitches sung by the live musician. This understanding of pitch within a harmonic context is vital to the cognitive processes described in Section 5 of this paper. The rules upon which we base our cognition system assume that the musical input is tonal, so our organizational scheme will facilitate the cognitive processes necessary to infer emotional meaning.

Currently, we are focusing our efforts on the perception of sung vocal melody, as it is the feature that we wish to highlight most prominently through the interaction between the performer and the virtual character.

To organize vocal input within a harmonic context, our system incorporates aspects of an existing music-theoretical melody encoding system devised by Deutsch and Feroe [4]. Their encoding system must, of course, be adapted for our needs. The real-time nature of our system makes our encoding task different from one based on a traditional harmonic analysis. We do not have the ability

to assess an entire musical work at once, but rather must operate in a linear fashion as the musical input arrives.

Deutsch and Feroe describe the representation of absolute pitch values (which we extract from vocalization and keyboard input using Max/MSP patches) as the lowest level of musical information representation. They propose a higher-level system which integrates the raw pitch information into the tonal context within which it is contained. Their system assesses a musical phrase to identify an appropriate pitch *alphabet* (chromatic scale, diatonic scale, etc...) which contains each element of the phrase to be described. They choose a dominant event in the phrase to use as a *reference element*. Each note in the phrase is then described in terms of its position in the specified alphabet with relation to the reference element. Additionally, their model allows complex or repetitive phrases to be described in a hierarchical fashion.

We have complied with their notion of an alphabet, a reference element, and the specification of all other elements in terms of their relationship to the reference element. Our system does not encompass all the features of Deutsch and Feroe’s model, as it is merely a subset of their extensive music theoretical model. Instead, it takes a more basic approach with the intent that it could be expanded in the future. One way in which our approach must necessarily deviate from Deutsch and Feroe’s is that, while their assessment of a dominant element in a melodic phrase benefits from the ability to make this determination after-the-fact, our system must address and encode melodic information as it is performed live, with no previous knowledge of the score. For this reason, we have chosen to use the tonic note of the key signature of the piece as the reference note. All further notes perceived are encoded as they are related to the tonic note of the scale.

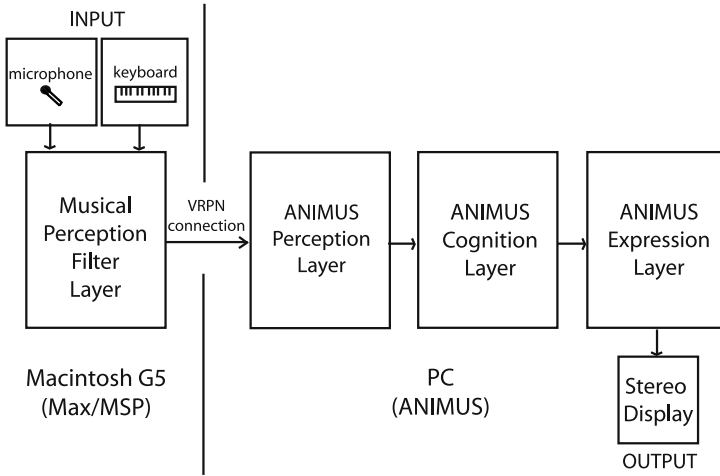
- **Example:** Within the key of C, the tonic note, **C**, would be the *reference note*. If the chosen *alphabet* was the chromatic scale (a scale which contains 12 semitones), **D** (a major second above C) would be represented as being *two steps in the alphabet above the tonic reference note*, while **Eb** (a minor third above C) would be *three steps in the alphabet above the tonic reference note*.

As will be discussed further in Section 5, encoding vocal input within its harmonic context will help to simplify the process of relating associated cognitive meaning to intervallic relationships within the melody line.

Although we are not identifying a dominant musical event within each phrase in order to aid in melody encoding, we are still interested in trying to assess the importance of each sung note in the phrase. In order to determine which notes are emphasized by the performer, we choose to characterize emphasized notes as those which can be differentiated from notes in their surroundings due to increased volume (volume is described by Cooke [2] as a “vitalizing agent”<sup>3</sup> which implies emphasis) or a sudden shift in register (notes significantly higher or lower

---

<sup>3</sup> Cooke, *The Language of Music*, p.94.

**Fig. 3.** System Architecture

than their surrounding notes). Our system calculates a suggested ‘importance’ value for each sung note.

We must then communicate these extracted events to the PC running the ANIMUS engine via a network connection (see Figure 3 for a system diagram). We have integrated pieces of the Virtual Reality Peripheral Network (VRPN) [13] into the Max environment by custom-creating our own Max external object, **vrpnserver**. This object takes the extracted features as input, and runs a VRPN server on the Macintosh. The PC running the ANIMUS engine’s perception layer can connect to **vrpnserver** as a client in order to access the extracted musical feature data. This data is then entered on the ANIMUS blackboard. ANIMUS characters monitor this blackboard to obtain information about the musical performance in order to carry out the cognitive and expressive tasks required to simulate responsive behaviour.

## 5 Simulating an Emotional Response in the Cognition Layer

In the cognition layer, the information sent by the perception layer must be received and analyzed in order to simulate the internal emotional state of the virtual character. The ANIMUS blackboard contains the information parsed from the live musical performance (sung intervals, data regarding the importance of each sung note, information about the singer’s vocal timbre, and chord data describing what the user is playing).

We must then simulate a cognitive awareness of the perceived data in order for the ANIMUS character to have an interesting internal state which it may then express through animated behaviours.



Cognitive awareness of musical performance could take many forms. Characters could assign feelings of happiness to a particular melody that they find pleasing, or they could enter a fearful state after perceiving a certain series of chords which they find threatening. Characters could dislike the piercing soprano of the Queen of the Night's coloratura, or express admiration for the rich mezzo tones of Carmen's "Habañera."

In order to create an interesting and flexible cognitive layer for our ANIMUS character, we have chosen to implement aspects of Deryck Cooke's research as described in "The Language of Music" [2]. Cooke's study of a widespread assortment of classical works provides insight into a more generalized relationship between music and emotion. Instead of implementing a character which enjoys one specific melody and dislikes another, we are interested in creating a character which is flexible enough to simulate an emotional response to music-theoretical features within a melody line. Cooke has discerned certain features to be salient features within a large number of musical pieces. He theorizes that certain features used in Western tonal music represent particular emotional concepts in a relatively universal way.

Cooke identifies "the basic expressive functions of all twelve notes of our scale."<sup>4</sup> If a melody contains many instances of the minor third, Cooke's theory states that the proper interpretation of the passage would be "stoic acceptance" or "tragedy".<sup>5</sup> He bases this inference upon many cited examples of musical passages expressive of tragic emotion which contain the minor third, such as Violetta's deathbed scene from Verdi's "La Traviata". Conversely, Cooke cites the American folk song "Polly-wolly-doodle" to exemplify how a major third often signifies "concord" or "joy".<sup>6</sup>

By applying his rules to sung vocal melodies, we can assign cognitive meaning to elements of the live musical performance. As noted in Section 4, our Musical Perception Filter Layer describes all sung melody notes by their tonal context within the existing key signature. Knowing this information, we can easily link each sung pitch to the emotional context extracted by Cooke. This emotional context can then serve to modify the ANIMUS character's internal state and trigger responsive behaviour.

The ANIMUS system uses a 'driver system' to control character behaviour. Familiar to many computer game users, due to its use in "The Sims" [5], a driver system operates on the principle that once the level of a specific character feature reaches a target maximum or minimum, behaviour is triggered. Sims fall asleep on their feet when their energy driver reaches its minimum, but get up out of their beds when their energy driver is at its maximum value. Similarly, our ANIMUS characters can use driver systems to express their emotional state.

According to Cooke's theories, a minor third signifies tragedy, while a major third signifies joy. Our ANIMUS character may have a 'happiness driver', the level of which is increased if the singer sings a melody which contains many

---

<sup>4</sup> Cooke, *The Language of Music*, p.89.

<sup>5</sup> Cooke, *The Language of Music*, p.90.

<sup>6</sup> Cooke, *The Language of Music*, p.90.

instances of major thirds, and decreases if a minor third is sung. Since our perception layer also assigns an ‘importance’ value to each sung note, the amount by which the ‘happiness driver’ is increased or decreased may also be dependent upon the importance value of the sung notes.

The other extracted musical features (chords, vocal timbre, etc...) can also be linked to the ANIMUS character drivers in order to influence character response.

The cognitive processing of musical input allows the ANIMUS character to maintain a fluctuating emotional state during the course of a live musical performance. This emotional state is then conveyed to the audience via the ANIMUS expression layer, which uses animations to visualize character behaviour.

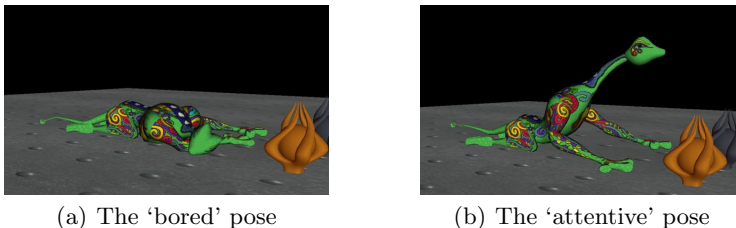
## 6 Visualizing Character Emotions Through Movement

An ANIMUS character is animated using keyframe-based interpolation. An ANIMUS character is a three-dimensional model that has a controllable skeleton. A variety of endpoint skeletal poses are defined using three-dimensional modelling software. All intermediate poses are generated at run-time in order to generate fluid transitions between these endpoint poses. This allows the animations to be dynamic, a key feature of the ANIMUS engine.

The ANIMUS expression engine allows the designer to define character animations both in terms of which keyframe poses are used to create a motion, and the speed of the transition between these specified poses.

These design decisions can then be linked to the ANIMUS character’s cognitive state. For example, Taylor, Torres, and Boulanger describe an ANIMUS character who is attentive to vocalizations within his environment [12]. When the user of the system is silent, the character’s cognition layer registers a high value in his ‘boredom’ driver. His expression layer translates this internal state by displaying him slumped in a ‘bored’ pose (see Figure 4a). When information about sung vocalizations reaches his perception layer, his cognition layer decreases its ‘boredom’ driver. His expression layer responds by rapidly transitioning him from his ‘bored’ pose to his ‘attentive’ pose, adjusting the position of his head so that he looks towards the perceived source of the sound (see Figure 4b).

We intend to enhance these simple animations by using the extracted emotive properties from a live musical performance as parameters which modify the portrayed ‘mood’ of the character’s actions. Using Cooke’s theories to infer an



**Fig. 4.** A posed ANIMUS character

emotional context from a musical performance, extracted emotional indicators can be used to assist in selecting appropriate keyframe poses and transition rates when creating the character's animations.

As an example, consider the case of a 'walking' animation. If the emotional context of a musical passage is inferred to be 'joyful,' keyframe poses can be selected in which the character appears to stand tall and energetic, and transitions between keyframe poses can occur in a smooth and rapid fashion. The character appears to walk with a brisk and confident stride. If the emotional context of the musician's performance begins to reflect 'despair', the keyframe poses can be selected to slump the character's shoulders. Transitions between the poses can slow down in order to reflect a sad and halting step.

## 7 Conclusion and Future Work

We have described a method of music visualization which provides a visual interpretation of a musical performance's emotional content by modifying the behaviour of a virtual character. We have chosen to base the character's cognitive understanding of emotional content upon the theories of Deryck Cooke.

Currently, we are working on completing our implementation of the character's cognitive layer in order to create an interesting and 'believable' virtual character. We would like to begin collaborating with a visual artist to create a sophisticated virtual character with a wide library of poses, so that we may have greater flexibility within the ANIMUS expression layer to create dynamic animations, evocative of a wide range of character emotions.

When our development of the cognitive and expression layers is complete, we hope that this system could be used in a live performance setting. The visual dialogue between the live performer and the virtual character enriches the musical experience. The virtual character's responses add a visual component that illustrates the emotive capacity of music. We would like to explore the interaction between the human musician and the animated character through an audiovisual piece composed in tandem by a musician and a visual artist.

We believe that this system represents a novel way to illustrate a 'human-like' perceptual and cognitive understanding of the emotive capacity of music.

## Acknowledgments

The use of the VRPN library was made possible by the NIH National Research Resource in Molecular Graphics and Microscopy at the University of North Carolina at Chapel Hill.

## References

1. The ANIMUS Project: A Framework for the Creation of Emotional and Rational Social Agents. <http://www.cs.ualberta.ca/~dtorres/projects/animus>  
Computing Science Department, University of Alberta.

2. Cooke, D. *The Language of Music*. New York: Oxford University Press, 1959.
3. Cycling '74. *Max/MSP*.
4. Deutsch, D. and Feroe, J. The Internal Representation of Pitch Sequences in Tonal Music. *Psychological Review*, 88, pages 503-522, 1981.
5. Electronic Arts. *The Sims*.
6. Koelsch, S., Gunter, T., Friederici, A.D. and Schroger, J. Brain indices of music processing: "Nonmusicians" are Musical. *Cognitive Neuroscience*, 12, pages 520-541, 2000.
7. Levin, G. and Lieberman, Z. In-situ Speech Visualization in Real-Time Interactive Installation and Performance. In *Proceedings of the 3rd International Symposium on Non-Photorealistic Animation and Rendering*, pages 7-14. ACM Press, 2004.
8. Oliver, W., Yu, J. and Metois, E. The Singing Tree: Design of an Interactive Musical Interface. In *DIS'97: Proceedings of the Conference on Designing Interactive Systems: Processes, Practices, Methods and Techniques*, pages 261-264. ACM Press, 1997.
9. Ox, J. Two Performances in the 21st Century Virtual Color Organ. In *Proceedings of the Fourth Conference on Creativity and Cognition*, pages 20-24. ACM Press, 2002.
10. Patel, A.D., Gibson, E., Ratner, J., Besson, M. and Holcomb, P.J. Processing Syntactic Relations in Language and Music: An Event-Related Potential Study. *Journal of Cognitive Neuroscience*, 10, pages 717-733, 1998.
11. Puckette, M., Apel, T. and Zicarelli, D. Real-Time Audio Analysis Tools for Pd and MSP. In *Proceedings of the International Computer Music Conference*, pages 109-112. International Computer Music Association, 1998.
12. Taylor, R., Torres, D. and Boulanger, P. Using Music to Interact with a Virtual Character. In *Proceedings of New Interfaces for Musical Expression*, pages 220-223, 2005.
13. Taylor II, R.M., Hudson, T.C., Seeger, A., Weber, H., Juliano, J. and Helser, A.T. VRPN: A Device-Independent, Network Transparent VR Peripheral System. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 55-61. ACM Press, 2001.
14. Torres, D. and Boulanger, P. A Perception and Selective Attention System for Synthetic Creatures. In *Proceedings of the Third International Symposium On Smart Graphics*, pages 141-150, 2003.
15. Torres, D. and Boulanger, P. The ANIMUS Project: A Framework for the Creation of Interactive Creatures in Immersed Environments. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, pages 91-99. ACM Press, 2003.

# Knowledge in the Loop: Semantics Representation for Multimodal Simulative Environments

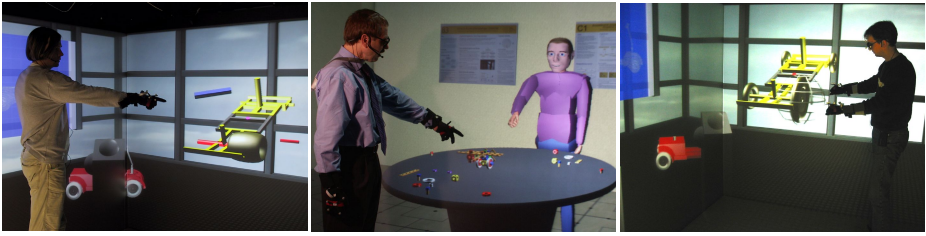
Marc Erich Latoschik, Peter Biermann, and Ipke Wachsmuth

AI & VR Lab, University of Bielefeld

**Abstract.** This article describes the integration of knowledge based techniques into simulative Virtual Reality (VR) applications. The approach is motivated using multimodal Virtual Construction as an example domain. An abstract Knowledge Representation Layer (KRL) is proposed which is expressive enough to define all necessary data for diverse simulation tasks and which additionally provides a base formalism for the integration of Artificial Intelligence (AI) representations. The KRL supports two different implementation methods. The first method uses XSLT processing to transform the external KRL format into the representation formats of the diverse target systems. The second method implements the KRL using a functionally extendable semantic network. The semantic net library is tailored for real time simulation systems where it interconnects the required simulation modules and establishes access to the knowledge representations inside the simulation loop. The KRL promotes a novel object model for simulated objects called *Semantic Entities* which provides a uniform access to the KRL and which allows extensive system modularization. The KRL approach is demonstrated in two simulation areas. First, a generalized scene graph representation is presented which introduces an abstract definition and implementation of geometric node interrelations. It supports scene and application structures which can not be expressed using common scene hierarchies or field route concepts. Second, the KRL's expressiveness is demonstrated in the design of multimodal interactions. Here, the KRL defines the knowledge particularly required during the semantic analysis of multimodal user utterances.

## 1 Introduction

VR tools have historically been built around the graphics system as the major simulation module. Nowadays, they additionally support the VR design process with a variety of features. These include support for application or behavior graphs via field-routing, input and output device abstraction, networking as well as scripting capabilities. Specifically required features, like a thorough or approximated physical simulation or gesture and speech processing for novel—multimodal—interaction methods (see Fig 1), have to be integrated using the extension methods provided by the VR and simulation tools utilized. Here, a promising research direction strives for a conceptual combination of VR and AI related methods to support the design of Intelligent Virtual Environments (IVEs) (Luck & Aylett, 2000).



**Fig. 1.** Multimodal interaction in knowledge supported construction tasks. Left: A user selects a wheel and connects it to a complex chassis (“*Take [pointing gesture] this wheel and connect it there...*”) (Latoschik, 2001). Middle: The user and the artificial communication partner MAX agree on the referenced part on the table (Kopp *et al.*, 2003). Right: The user scales a previously connected part which keeps its attributes, here the wheel’s roundness (Biermann & Jung, 2004).

AI and VR concepts have been combined in a variety of research projects using customized and application specific integration methods. In the last few years, several approaches have aimed at a more general way of a conceptual and technical integration of AI techniques into simulation based systems (Cavazza & Palmer, 2000; Luck & Aylett, 2000; Peters & Shrobe, 2003; Soto & Allongue, 2002). An integration based on a common representation layer as proposed in (Cavazza & Palmer, 2000) offers several advantages regarding adaptability and reusability. Its content can be made persistent using an external file format where AI, graphics, physics and other simulation related content are coequally expressed in a common format.

A fundamental integration of AI and VR provides the potential for a wide range of possible applications including heuristic approximations of—e.g., physical—simulation features and advanced multimodal interaction setups. For example, we have enriched our VEs with multimodal instruction type interactions as well as with a humanoid communication partner called MAX (Kopp *et al.*, 2003). Here, lexical and semantic content about the simulated scene is mandatory during both, the analysis of the user’s and the generation of Max’s multimodal utterances.

### 1.1 Knowledge Representation in VR

This article presents a unified and comprehensive method of knowledge integration and access in VR-based simulation systems. This method will be illustrated in two example areas: knowledge supported virtual construction and multimodal interaction. On shallow examination the two areas seem to differ significantly. In a closer examination, we will illustrate how necessary generalizations made from both, the example applications’ data representations, as well as from the VR-simulation specific data representations lead to a common knowledge layer capable of a high-level description of advanced simulation features.

We start by introducing the used terminology and by a closer analysis of the built-in semantics of existing representations of generic VR-simulation system. From a knowledge based view, objects and structures of a simulation system can be defined by *entities*, *attributes*, *relations*, and *concepts*.

Entities represent simulated objects. Attributes describe certain feature-value pairs of entity representations. For example, simple well known attributes for the graphics part are (RGBA) diffuse colors of the utilized lighting model or 9 float values for transformation specification as illustrated in the upcoming Fig. 2 and Fig. 3 by the 9DOF nodes. Each of these nodes carries 9 floats which are associated to a given entity, e.g., `2_hole_rod` or `hole1` in Fig. 2, to specify its transformation. All atomic data about entity features are eventually described as attributes, from simple color values, diameters of holes, entity masses up to functional features attributes like *is-scalable* or *is-connectable*.

Relations generally describe n-ary predicates of attributes, entities and concepts. Concepts represent categories, semantic circumscriptions of objects and attributes of the regarded target domain as illustrated in Fig. 2 and Fig. 4 for the base node of the `2_hole_rod` entity which instantiates a **ROD** concept. Concepts are attribute and entity descriptions, patterns or *classes*. A typical known relation of scene graph systems is the *part\_of* relation that defines the scene hierarchy and grouping behavior.

With respect to the example domain, the term *part* denotes non-decomposable but modifiable entities used in the construction process. Parts consist of *sub-parts* which relate to semantically self-contained sections of parts. A sub-part is defined (1) by its position relative to a part's frame of reference and (2) by a set of sub-part specific attributes which describe the sub-part's type. Sub-parts can not be instantiated without a part to which they are bound conceptually– they can not be deconstructed during user interaction.

## 1.2 Built-In Semantics of VR-Simulation Systems

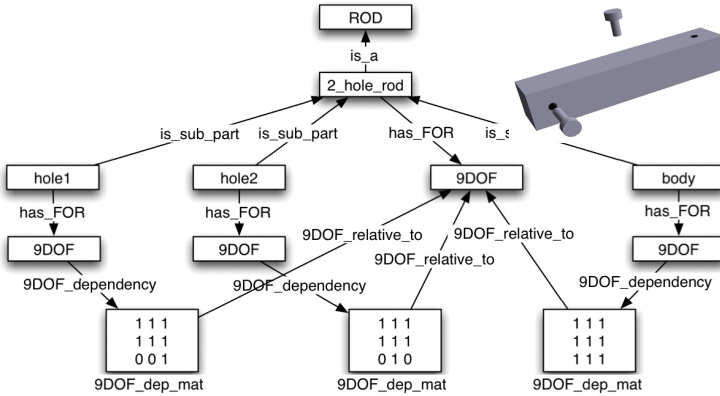
The semantics of attributes and relations commonly used in simulation systems is defined procedurally. The interpretation of attributes and relations is defined by the underlying technical processes of the simulation modules. For example, color-attribute values are used to calculate pixel-shadings with respect to utilized lighting model, and *part\_of* scene graph relations define the accumulative multiplication of matrices on the matrix stack. This operational and fixed semantics limits the representations available to the predefined rendering tasks and complicates or even inhibits their utilization for different application specific representations. The expressiveness of the scene graph related *part\_of* relation as well as those of application graphs built from field route connections is strictly defined by the procedural semantics of the simulation system.

As a consequence, additional representations are necessary for reasonably complex applications since the existing features of the simulation engines are not expressive enough. This often results in purpose-built solutions which lose the declarative expressiveness, the reusability as well as the flexibility of the representation methods provided by the simulation engines. For example, a common solution in field route based systems is to define new node types which receive certain data and manipulate this data using specialized algorithms. Most VR-based simulation systems include methods for such extensions, e.g., VRML (Carey *et al.*, 1997) supports this approach using the built-in PROTO feature.

In the worst case, none of the built-in extension features are expressive and powerful enough for some applications. In that case, external modules are often loosely coupled to the simulation system and data is exchanged between them using standard interprocess communication (IPC) facilities. This in turn requires special purpose external synchronization and data-replication which complicates application development significantly or even prevents expandability and reusability of systems' components. Furthermore, in certain areas a loose coupling can in fact be insufficient. For example, continuous user interaction, e.g., dragging of an object, usually requires a high responsiveness which can not be guaranteed at all times using loose coupling without concurrently using IPC blocking behavior.

## 2 Simulation Knowledge Representation Layer

Our goal is a common Knowledge Representation Layer (KRL) which contains VR-simulation specific as well as application tailored knowledge. The subsequent explanations presuppose a simulation system which at least provides scene and behavior graph structures as for instance offered by the AVANGO toolkit (Tramberend, 1999). Hence, relations defined in the knowledge layer first of all have to represent the target system's representational features, and similar steps are necessary for other target systems.



**Fig. 2.** Knowledge representation of an example part (a rod with two holes, see upper right) which supports intelligent scaling operations

**sg\_part\_of** (scene graph parent/child): Transitive directed relation denoting grouping behavior of scene graph nodes. Implies accumulative multiplication of existent matrix attributes found at nodes followed in the given relation direction.

**fr\_connect\_to** (field route connection): Transitive directed relation denoting value propagation of attributes in the given relation direction.

**f\_control\_to** (field reference): Directed relation denoting referential (not routed) read/write access to fields of target nodes by specialized source nodes.



Additional relations are defined to express the application specific knowledge. The following relations and their semantics support virtual construction tasks that group parts to aggregates and specify geometric dependencies between (sub-)parts:

**is\_sub\_part**: Transitive directed relation denoting the association of a sub-part to a part.

**is\_a**: Transitive directed relation denoting a subsumption hierarchy for part concepts. **is\_a** implies inheritance behavior of certain attributes, e.g., lexical information, of parent concepts.

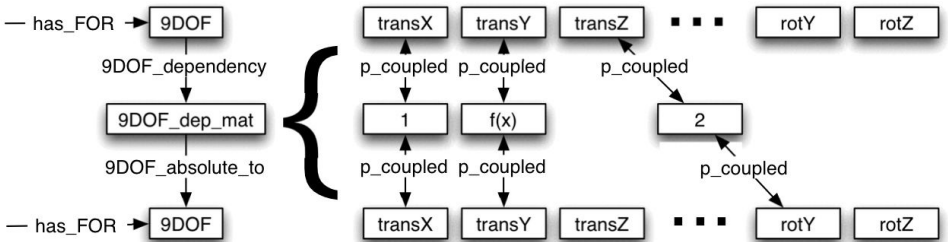
**has\_FOR**: Directed relation denoting the association of a frame of reference (FOR) to a given concept, e.g., a part or a sub-part.

**9DOF\_dependency**: Transitive directed relation denoting a geometric dependency between two 9DOFs (9 Degrees of Freedom) as parameterized by a **9DOF\_dep\_mat** concept which defines the dependencies (see following sections).

**9DOF\_relative\_to**: Transitive directed relation denoting the relative position, orientation, and scaling of a 9DOF with respect to a given 9DOF frame of reference.

Fig. 2 illustrates a segment of the resulting knowledge structure which supports intelligent scaling operations for a rod with two holes as sub-parts which are defined to behave independently during a scaling operation of the main part, e.g., to maintain the holes' roundness during scaling operations. All parts and sub-parts have associated 9DOF frames of reference which define their position, orientation and scaling using the **has\_FOR** relation. This ensures that grouping and transformation are expressed independently from each other. The sub-parts' FORs are defined to be relative to the main part's FOR via a dependent mapping defined by the **9DOF\_dependency** which parameterizes the **9DOF\_relative\_to** relation using the **9DOF\_dep\_mats**.

The semantics of this representation is as follows: The 3x3 dependency matrix of a **9DOF\_dep\_mat** defines how the factors for position (first matrix row entries), rotation (second row) and scaling (third row) are concatenated following the algebraic semantics of the **9DOF\_relative\_to** relation. In its plain assertion between two FORs, the **9DOF\_relative\_to** defines well known multiplication of homogenous coordinate representations which would express common scene-graph behavior. In contrast, the chosen representation allows an extensive parameterization of the concatenation type of two linked FORs. Fig. 3 illustrates how arbitrary



**Fig. 3.** Parameterized coupling of attribute values which are calculated according to the algebraic rule as defined by the embracing relation, here the **9DOF\_absolute\_to** relation

parameters—constant values as well as functions—can be defined to modulate the algebraic effect or calculation rule of the active relation which couples two attributes.

The free interconnection of attributes even allows coupling between different geometric attributes or DOFs e.g., to have one part to rotate if another part translates or to match one part's scaling by a rotation of two other parts if two dependency matrices are used.

The two zeroes in the last row of the left and the middle 9DOF\_dep\_mat in Fig. 2. represent a missing `p_coupled` relation and hence define partial blocking of the 9DOF\_relative\_to semantics which defines a parent-child relation between the main part and the holes. This suppresses the consecutive impact of parent part's total scaling and only scales `hole1` in the z- and `hole2` in the y-direction (the principal axes of the holes' main directions).

### 3 Interaction Knowledge Representation Layer

The KRL is not limited to the representation of geometric dependencies as motivated for the virtual construction task. Its overall goal is to support application specific representation models as well as commonly required VR-related modeling tasks. This includes high level representations of entity data and structures for the variety of involved software modules, e.g., for the graphics, physics and the interaction components.

The idea of a semantic representation is in fact strongly inspired by the intention to utilize multimodal—gesture and speech driven—interactions in VEs. Processing of multimodal utterances can be roughly divided into several phases: Speech and gesture detection, semantic speech and gesture analysis, multimodal integration and pragmatic analysis. During processing, several of these phases frequently access semantic content from redundant data representations of other simulation modules. Here, a unified KRL partly solves the multiple database problem.

A major design goal for the knowledge representation layer is to support semantics, necessary during interaction. This includes, e.g., mappings between conceptual and lexical data for a variety of concepts and attributes. These concepts do not only represent perceivable entities and their features in the artificial world but also abstract concepts, e.g., holes in a part or actions a user can carry out.

The necessity for the representation of semantics is evident for the semantic analysis phase which has to map lexical and gestural expressions to conceptual knowledge. Fig. 4 presents another view into the semantic network which represents the example two-holed rod. The grey nodes illustrate instantiation of a two-holed rod. The conceptual definition of a two-holed rod (grey rectangular nodes) is used as a template for the actual instance (grey oval nodes). Instances represent the existing entities of the simulation. Their inheritance of concepts and attributes as defined by their conceptual templates is customizable. If they are not inherited during instantiation, their inheritance behavior is defined per relation following the connecting relations (here `inst_of` and `is_a`). Specialized negating relations (pattern `<x>not_<y>`, e.g., `hasnot_feature`) provide a method to locally override the default inheritance behavior.

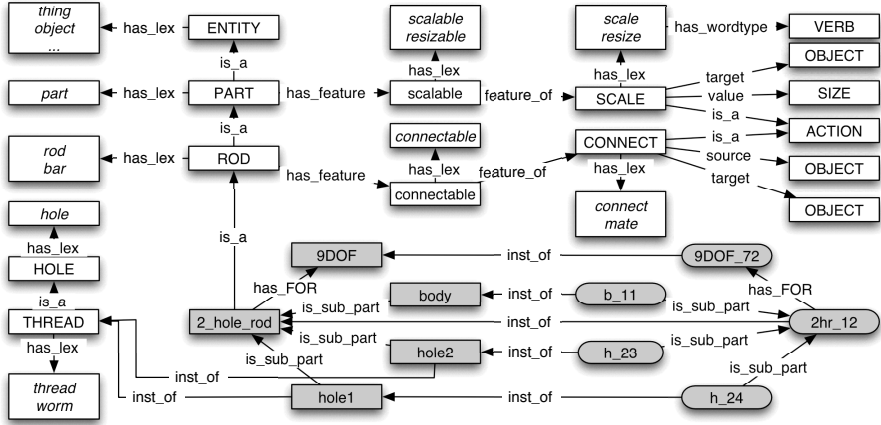


Fig. 4. Conceptual and lexical knowledge representation of the two-holed rod

Fig. 4 illustrates the interconnection between entity features which define certain simulation behavior, e.g., whether an entity is scalable, and the representation of the according user action by the **SCALE**, **CONNECT** and **ACTION** concepts. These interaction concepts further define their required conceptual knowledge to be fulfilled, e.g., a required target or a required interaction parameter. Where linguistic knowledge is given by linking concepts to lexical counterparts, the semantic analysis processing module is automatically enabled to map lexical information to conceptual knowledge which then can be instantiated during the processing.

For example, a connection of the screw and the two-holed rod in the virtual construction task can be accomplished in two ways. The direct manipulation way is to drag the screw. When the screw collides with the rod, the best fitting ports (here the holes) are heuristically determined and appropriate modules will be activated to simulate the connection (see subsequent sections). The second way is by using multimodal interaction, e.g., by saying: “Put [pointing gesture] that screw in [pointing gesture] this hole.” (see, e.g., Fig. 1. left). Focusing on the linguistic part, input is analyzed by mapping the incoming words to the lexicon which is defined by the target concepts of the *has\_lex* relation in Fig. 4. By backward traversing these relations during the parse process (Latoschik, 2002), the matching base concepts are retrieved from the KRL.

The conceptual knowledge is now used according to its semantics. For example, verbs found in imperative speech such as “Give me a...” will be looked up in the lexicon. Traversing the *has\_lex* relation will retrieve the respective concept. If this concept is then identified as an **ACTION**, e.g., by traversing the type hierarchy, matching interaction frames will be retrieved and instantiated which define required interaction information, e.g., the required target **OBJECT** instance(s). In addition to the illustration in Fig. 4, the actual existing knowledge base decomposes **OBJECT** concepts into their substructures, relates them to the two existing type hierarchies, and augments the linguistic structures with syntactic information.

These interaction frames, their concepts and attributes, are used as templates which are filled during input processing. For example, if the template requires one or more objects, the parse process will feed a reference resolving engine with the conceptual information necessary to identify one or more target entities (Pfeiffer & Latoschik, 2004). Since the KRL interconnects instances with the data representations of other simulation modules, this data, e.g., a quantitative RGBA value for a color attribute will be processed in the same way. Finally, completed interaction frames trigger the desired interaction.

## 4 Implementing and Applying the KRL

To implement the KRL, a knowledge representation tool, FESN (Functionally Extendable Semantic Network) (Latoschik & Schilling, 2003), has been developed. The FESN offers a specialized and adaptable semantic net formalism which is implemented as a C++ library and which has special features targeted at its utilization in VR simulation systems and for diverse application specific representation tasks.

**Attribute augmentation of concepts:** FESN nodes can carry additional attributes and values which allows a seamless transformation of a target system's representations into the FESN.

**Functional extensibility:** Flexible relation semantic. New relation types can be added easily. The semantics of relations is expressed by functions added to the relations.

**Built-In event system:** Changes of attribute values and the network's structure are monitored to enable automatic processing of changes submitted by simulation modules.

**Built-In event filter:** Concepts (nodes) of the FESN can be associated with multiple external attribute sources of the same attribute. A parameterized filter concept allows automatic evaluation and selection of concurrent—possibly conflicting—value changes.

**External XML representation:** The FESN supports SNIL, the Semantic Net Interchange Language, as an XML based external representation. This provides a convenient way to define and modify knowledge layer content (see Fig. 5).

Furthermore, the low level implementation of the FESN as a C++ library allows several performance optimizations which conserve processing resources in contrast to commonly found high-level (e.g. PROLOG or LISP based) semantic net realizations. This is particularly important for the FESN's utilization in interactive real-time simulations.

Using the FESN as the base formalism, the KRL completely defines all data and knowledge required by a simulation system in a unified representation, including graphics, physics, audio or even AI and interaction content. The KRL's content is applied to simulation systems in two ways. The first, uncoupled, method transforms knowledge bases defined by SNIL into representations compatible with the simulation

modules of the target system. Several of these modules support external representations. Some of them support proprietary XML based formats, e.g., we have previously developed an XML based format for a simulation module which handles variant parts: VPML (Variant Part Markup Language) (Biermann & Jung, 2004). The required mapping between source and target representation is conveniently achieved via XSLT processing where the mapping rules only have to be statically defined once.

In contrast to the uncoupled method, the coupled method embeds the FESN as a separate module directly into the simulation system's process space and latches knowledge access into the simulation loop(s) of all simulation modules which share this process space. In such setups, the FESN acts as a central knowledge base and monitoring instance. Its event system provides a method to control and filter value changes of attributes which might be proposed by several modules concurrently, e.g., for the 9DOF attribute.

#### 4.1 Semantic Entities

The coupled method provides two ways of knowledge access: First, arbitrary FESN queries can be directly called on semantic net structures which are mapped into the process space of the calling modules. Second, modules can use an object centered KRL-access to dedicated entities in their own proprietary object representation. In effect, each of the simulation module specific entities has a corresponding counterpart represented in the KRL. By augmenting the module specific entity representation with an FESN-interface—for example, using object oriented multiple inheritance schemes—the entities' semantic data is directly accessible by the given simulation module. This architecture leads to a novel object or entity model we call **Semantic Entities**. Semantic Entities link proprietary entity representations with the correspondent instances of the knowledge layer and provide a standardized way of accessing the KRL.

The uniform KRL-access via Semantic Entities allows for an increased and powerful modularization of simulation systems. Typical architectures for implementing some specific functionality for a given simulation utilize so-called engines. Engines are dedicated modules in scene graph based systems which implement a certain function and which apply their results by accessing target nodes and attributes directly or by propagating attribute changes using a behavior graph. Here, target nodes and attribute-routes have to be specified by the application programmer specifically, be it for a simple interpolator engine or for an advanced collision and dynamics engine. Using Semantic Entities, information about object animation and manipulation specification is conveniently defined declaratively using the KRL. By querying the KRL via the Semantic Entities in the engines' or modules' process space, requested object features, like if an object is movable, or collidable, or any other feature a specific engine requires, can directly be accessed.

For example, we have developed several components for multimodal interaction which completely rely on Semantic Entity access. Dedicated engines preselect and sort possible targets' Semantic Entities in the scene representation according to users' gestures like view and pointing direction. This set of possible targets is then further

restricted to objects which satisfy the semantic interpretation of type or attribute specifications, e.g., during the processing of definite noun phrases like "...the blue rod...": A KRL retrieval for the word "...rod..." will find the ROD concept by following its `has_lex` relation in Fig. 4. This result will restrict the preselected set of Semantic Entities to entities of the required type, e.g., to include the `2hr_12` instance in Fig. 4 which is of type ROD following the transitive `inst_of` and `is_a` relation. The semantic interpretation engine uses Semantic Entities for object centered KRL access to map the utterance's meaning to the semantic scene description which includes representations of the user and possible interaction partners as in Fig. 1. Other components evaluate the construction specific knowledge of entities and automatically instantiate required simulation modules which implement a given entity feature as will be explained in the following sections.

The Semantic Entity object model greatly simplifies the development of Virtual Environments. It promotes modularization, easy adjustment, and reuse of simulation components. Components access specific entity features via Semantic Entities. Additionally, the components themselves can be represented in the KRL to provide an automatic mutual matching of tools and target objects since they share the same representation.

## 4.2 Mapping Knowledge to Target Systems

Besides KRL-access, both knowledge application methods have to map the FESN representations to a target system's internal representation. This mapping transforms the modeled knowledge into structures readable and interpretable by the respective simulation modules. For the example two-holed rod, the knowledge fragment that defines the geometric dependency of `hole1` (see Fig. 2. ) is illustrated in Fig. 5 using the SNIL notation.

Our current AVANGO-based target system supports a scene graph metaphor with a field route concept and allows implementation of new proprietary node types. This justifies grouping of sub-parts as children of the main part. But this maps the modulated `9DOF_relative_to` relations to fixed `sg_part_of` relations. To overcome the fixed scene graph semantics, a new node type called Constraint Mediator (CM) was developed which is parameterized by a dependency matrix for constraint definition. Instead of Geometric Constraint Solvers as in (Fernando *et al.*, 2001), which solve the constraints in an external system, the CMs in our target system implement the defined geometric dependencies as constraint nodes in the scene graph and apply them directly to the 4x4 transformation matrices. In other possible target systems, like VRML97, these constraints could be realized by using Script-Nodes, whereas the implementation of the scripting interface in VRML is often very slow (Diehl & Keller, 2002).

The CM nodes can monitor several fields to restrict and propagate their values. Unlike normal field connections (e.g, the field routes in VRML), CMs can propagate field-values in both directions and can alter the values while propagating, to establish complex constraints directly in the scene graph. These constraints implement the geometric dependencies of the knowledge based simulation as defined by the knowledge layer in the target system.

```

<semantic-net>
  <node name="2_hole_rod" type="Default"/>
  <node name="9DOF_2_hole_rod" type="Default">
    <slot name="FOR" type="9DimVect"
      inheritanceType="Attribute"
      value="0 0 0 0 0 0 1 1 1"/>
  </node>
  ...
  <node name="hole1" type="Default"/>
  <relation typeName="is_sub_part">
    <start-node nodeName="hole1"/>
    <end-node nodeName="2_hole_rod"/>
  </relation>
  <node name="9DOF_hole1" type="Default">
    <slot name="FOR" type="9DimVect"
      inheritanceType="Attribute"
      value="-.2 0 0 0 0 0 0 1 1"/>
  </node>
  <relation typeName="has_FOR">
    <start-node nodeName="hole1"/>
    <end-node nodeName="9DOF_hole1"/>
  </relation>
  <node name="9DOF_dep_mat_hole1" type="Default">
    <slot name="FOR" type="81DimMat"
      inheritanceType="Attribute"
      value=" ... (9x6 identity)1
        0 0 0 0 0 0 0 0 0
        0 0 0 0 0 0 0 0 0
        0 0 0 0 0 0 0 0 1"/>
  </node>
  <relation typeName="9DOF_relative_to">
    <start-node nodeName="9DOF_dep_mat_hole1"/>
    <end-node nodeName="9DOF_2_hole_rod"/>
  </relation>
  ...
</semantic-net>

```

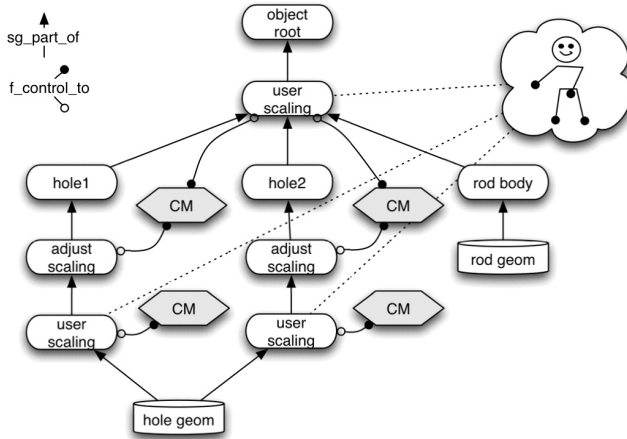
**Fig. 5.** SNIL fragment that partly describes the two-holed rod example. The extended dependency matrix that defines hole1's dependent scaling is defined as an attribute of the 9DOF\_dep\_mat\_hole1 concept.

#### 4.2.1 Inter-Part Constraints (Scaling / Transformations)

One application area for the dependency matrices is the simulation of building parts and sub-parts, which can have complex scaling behavior, depending on the scaling of their parent parts. Fig. 6 shows an example of a scene graph section with CMs which prevent the deformation of the holes when scaling the rod. When the user scales the rod in the X- or Y-direction, the two upper CMs in Fig. 6 set the adjust-scaling of the holes to the inverse of the scaling of these directions to maintain the size and roundness of the holes. When scaling in the Z-direction—which is the direction of the

<sup>1</sup> Only the lower three matrix rows which are relevant for scaling dependencies are depicted for readability.

main axis of the first hole—this hole is scaled with their parent, to fit with the thickness of the rod. The other two CMs restrict the user scaling of the holes to be equal in X- and Y-direction and to the identity scaling for the Z-direction. This allows the user to adjust the diameter of the holes with respect to the defined scaling behavior of the part.



**Fig. 6.** Scene graph section with embedded Constraint Mediators for sub-part to part dependent scaling. CMs implement dependency matrix semantics for scene graph systems.

The coupling of other transformations as inter-part constraints is also possible. The parameters of parametrical changeable parts which are described in the KRL can be linked. This concept of linked transformations allows the simulation of gears in the virtual environment (Biermann & Wachsmuth, 2004). These gears are realized using the Constraint Mediators for the coupling of the transformation parameters. Simple rotational gears can be generated by using a coupling of the two rotations of the corresponding sub-parts with a certain transmission factor. E.g., a coupling of rotational and translational parameters can lead to a pinion gear.

#### 4.2.2 Part-Part Constraints (Connections)

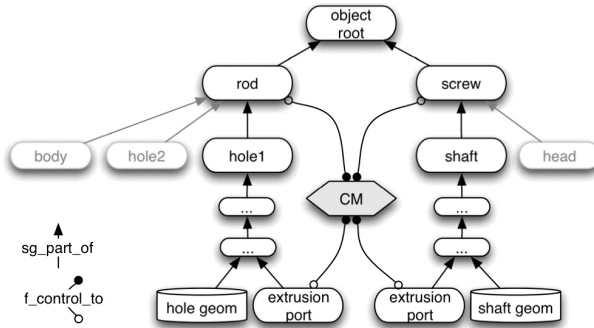
While the constraints for the scaling behaviour are normally fixed for each part, the simulation of part-part connections requires dynamic constraints. The coupling via constraints also allows the simulation of mating properties which can not be directly derived from the geometry of the parts. For example, it is possible to have plane-port connections, which restrict the movements of the parts so that the two connected planes always keep connected and in the same orientation, while they can slide until they reach the edges of the planes.

The mating geometries (so called Ports) define different degrees of freedom for the established connections. The knowledge base contains the information of the constraints for each Port-type. The restricted movements of the connected parts are also controlled by Constraint Mediators, which are established via the semantic net, when a new connection is established. Technically, a connection is implemented by



oriented mating points whose possible movements are restricted by CMs configured according to the Port types' constraints.

The example in Fig. 7 illustrates how the connection between the screw and one of the rod's holes is reflected in the target system's scene graph: A CM establishes the constraints, which simulate the connection of a screw fitted in a hole of a rod. The CM for the connection watches the positions of the two connected extrusion Ports and—in this case—alters the matrixes of the root nodes of the parts, if the positions of the Ports do not respect the constraints that are defined for this type of connection.



**Fig. 7.** Scene graph section for two parts interconnected by a constraint mediator to implement part-part geometric dependencies

## 5 Conclusion

We have introduced a general method for integrating semantic information into the VR simulation and interaction loop. It is based on an abstract knowledge representation layer (KRL) for high-level definition of complex application designs. The FESN, the KRL's base formalism, provides a convenient method for AI related application solutions. It interconnects the data structures of the required simulation modules and provides external representation formats to express simulation data as well as application logic in a human readable way and hence supports reusability and extensibility of once developed representations.

Semantic Entities as unified object models provide the necessary method to uniformly access the KRL during runtime. Using Semantic Entities as the central entity access facility provides several advantages: Simulation modules can be built which automatically match their functions to the respective target objects. Possible object actions and interactions can be specified in advance, using a declarative notation. Developed simulation components can be reused, adapted and modified easily. Even complex application developments can be performed by generating a few lines of XML code for the required knowledge structure.

The usefulness of a high-level knowledge representation has been demonstrated (1) for a generalized scene representation which extends the expressiveness of commonly used scene graphs and (2) for the implementation of novel multimodal interaction methods. The illustrated method is currently applied in several projects in our lab

which focus on multimodal human-computer interaction and virtual construction applications.

Future work expands the KRL to support a variety of simulation components from different graphics packages to physics libraries. The final goal is a platform which conceptually allows abstract definition of intelligent VR applications via the KRL with as minimal adaptations from the utilized simulation systems core functionality. This work has already begun with the development of an automatic data synchronization and replication framework required.

## Acknowledgement

This work is partially supported by the Deutsche Forschungsgemeinschaft (DFG).

## References

- Biermann, P., & Jung, B. (2004). Variant design in immersive virtual reality: A markup language for scalable CSG parts, *AMDO2004*: Springer.
- Biermann, P., & Wachsmuth, I. (2004). Non-physical simulation of gears and modifiable connections in virtual reality, *Proceedings Fifth Virtual Reality International Conference (VRIC 2003)* (pp. 159-164). Laval, France.
- Carey, R., Bell, G., & Marrin, C. (1997). *Iso/iec 14772-1:1997 virtual reality modeling language (vrml)*: The VRML Consortium Incorporated.
- Cavazza, M., & Palmer, I. (2000). High-level interpretation in dynamic virtual environments. *Applied Artificial Intelligence*, 14(1), 125-144.
- Diehl, S., & Keller, J. (2002). Constraints for 3D graphics on the internet, *Proceedings of 5th International Conference on Computer Graphics and Artificial Intelligence 31A'2002*. Limoges, France.
- Fernando, T., Marcelino, L., & Wimalaratne, P. (2001). Constraint-based immersive virtual environment for supporting assembly and maintenance task, *Human Computer Interaction International 2001*. New Orleans, USA.
- Kopp, S., Jung, B., Lessmann, N., & Wachsmuth, I. (2003). Max—a multimodal assistant in virtual reality construction. *KI-Künstliche Intelligenz*, 03(4), 11-17.
- Latoschik, M. E. (2001). A gesture processing framework for multimodal interaction in virtual reality. In A. Chalmers & V. Lalioti (Eds.), *AFRIGRAPH 2001, 1st International Conference on Computer Graphics, Virtual Reality and Visualisation in Africa* (pp. 95-100): ACM Press.
- Latoschik, M. E. (2002). Designing transition networks for multimodal VR-interactions using a markup language, *Fourth IEEE International Conference on Multimodal Interfaces ICMI'02* (pp. 411–416). Pittsburgh, Pennsylvania: IEEE Press.
- Latoschik, M. E., & Schilling, M. (2003). Incorporating VR databases into AI knowledge representations: A framework for intelligent graphics applications, *Sixth IASTED International Conference on Computer Graphics and Imaging* (pp. 79-84): ACTA Press.
- Luck, M., & Aylett, R. (2000). Applying artificial intelligence to virtual reality: Intelligent virtual environments. *Applied Artificial Intelligence*, 14(1), 3-32.

- Peters, S., & Shrobe, H. (2003). Using semantic networks for knowledge representation in an intelligent environment, *PerCom'03: 1st Annual IEEE International Conference on Pervasive Computing and Communications*. Ft. Worth, TX, USA: IEEE.
- Pfeiffer, T., & Latoschik, M. E. (2004). Resolving object references in multimodal dialogues for immersive virtual environments, *IEEE VR2004* (pp. 35-42). Chicago: IEEE.
- Soto, M., & Allongue, S. (2002). Modeling methods for reusable and interoperable virtual entities in multimedia virtual worlds. *Multimedia Tools Appl.*, 16(1-2), 161-177.
- Tramberend, H. (1999). Avocado: A distributed virtual reality framework, *1999 IEEE Virtual Reality Conference (VR99)* (pp. 14-21). Houston, Texas: IEEE.

# Virtual Camera Planning: A Survey

Marc Christie<sup>1</sup>, Rumesh Machap<sup>2</sup>, Jean-Marie Normand<sup>1</sup>, Patrick Olivier<sup>2</sup>,  
and Jonathan Pickering<sup>3</sup>

<sup>1</sup> University of Nantes

<sup>2</sup> University of Newcastle Upon Tyne

<sup>3</sup> University of York

**Abstract.** Modelling, animation and rendering has dominated research computer graphics yielding increasingly rich and realistic virtual worlds. The complexity, richness and quality of the virtual worlds are viewed through a single media that is a virtual camera. In order to properly convey information, whether related to the characters in a scene, the aesthetics of the composition or the emotional impact of the lighting, particular attention must be given to how the camera is positioned and moved. This paper presents an overview of automated camera planning techniques. After analyzing the requirements with respect to shot properties, we review the solution techniques and present a broad classification of existing approaches. We identify the principal shortcomings of existing techniques and propose a set of objectives for research into automated camera planning.

## 1 Introduction

At a very basic level one of the objectives of photography and cinematography is to capture and convey information. Deciding where to position a camera or how to move a camera necessarily raises questions as to what information is to be conveyed and how this will be achieved. We propose three levels of description for the properties of an image: geometric, perceptual and aesthetic. Geometric properties capture the absolute and relative screen position, orientation and sizes of objects. Perceptual properties refer to intermediate stages of the visual processing pipeline, for example, the occurrence of visual gestalts and other properties that impinge on our recognition of objects and their spatial relations with each other. Aesthetic properties relate to notions of shot composition and are typified by terms frequently used (but hard to algorithmically characterize) by artists and art scholars, for example, compositional balance and unity.

In transposing these notions to virtual environments, researchers have been working on approaches to assist and automate positioning and path planning for virtual cameras. A common approach is to invoke declarative techniques by which a user articulates the properties required in the shot (*e.g.* what should be on the screen, where on the screen, which vantage angle) and a solver computes a solution, set of solutions, or best approximation to a solution. To date most actual systems rely solely on geometric properties. This paper presents a survey

of virtual camera planning techniques, and we structure our review by referring to two criteria:

1. the expressivity of the set of properties *i.e.* the assumptions pertaining to the properties, the qualitative and quantitative characteristics as well as the range of possible properties;
2. the characteristics of the solving mechanisms (*e.g.* generality, optimisation, local-minima failure, and computational cost).

In Section 2 we present the principles of camera planning and cinematography as they apply to the use of real world cameras. Section 3 reviews the uses of the cameras in computer games, a demanding practical field of application, and in Section 4 we review existing research before concluding with our requirements for future research.

## 2 Camera Planning and Cinematography

Direct insight into the use of real-world cameras can be found in reports of photography and cinematography practice [1,22,21]. Cinematography encompasses a number of issues in addition to camera placement including shot composition, lighting design and staging (the positioning of actors and scene elements) and an understanding of the requirements of the editor. For fictional film and studio photography camera placement, lighting design and staging are highly interdependent. However, documentary cinematographers and photographers have little or no control of staging and we consider accounts of camera placement in cinematography within this context. Indeed, real-time camera planning in computer graphics applications (*e.g.* computer games) is analogous to documentary cinematography whereby coherent visual presentations of the state and behavior of scene elements must be presented to a viewer without direct modification to the position or orientation of the elements.

### 2.1 Camera Positioning

Whilst characterizations of cinematography practice demonstrate a degree of consensus as to best practice, there is considerable variation in its articulation. Accounts such as Arijon's [1] systematically classify components of a scene (*e.g.* according to the number of principal actors) and enumerate appropriate camera positions and shot constraints. Not surprisingly, Arijon's procedural description of camera placement has been cited as the motivation for a number of existing automatic camera planning systems. By contrast accounts such as Mascelli's [22] are a less prescriptive and formulate camera planning in terms of broader motivating principles, such as narrative, spatial and temporal continuity.

### 2.2 Shot Composition

Camera positioning ensures the general spatial arrangement of elements of the scene with respect to the camera, thereby placing a coarse constraint on the

composition of a shot. That is, the position (and lens selection) determines the class of shot that is achievable, which is typically classified accordingly to the proportion of the subject included in the shot as: close up (*e.g.* from the shoulders), close shot (*e.g.* from the waist), medium shot (*e.g.* from the knee), full shot (*e.g.* whole body) and long shot (*e.g.* from a distance). However, precise placement and orientation of the camera is critical to achieving the layout of the scene elements in shot — referred to as the *composition* of the shot.

Composition is variously characterized in terms of shot elements including lines, forms, masses, and (in the cases of action scenes) motion. In turn, shots are organized to achieve an appropriate (usually single) center of attention, appropriate eye scan, unity, and compositional balance (arrangements of shot elements that affords a subconsciously agreeable picture). As psychological notions these terms are problematic to characterize and the empirical investigation of visual aesthetics is very much in its infancy. However, the validity or significance of the notions themselves cannot be questioned; eye tracking studies have demonstrated significant differences between the viewing behavior of observers of subjectively agreed balanced and unbalanced (through artificial modification) works of art [24].

Scenes that comprise significant amounts of motion and action pose different problems for cinematographers and editors, although general heuristics such as the triangle principle, use of a line of action, and compositional rules can be extended to these more complex configurations. The challenge for camera planning is to algorithmically formulate these principles in a manner appropriate to the particular application to be addressed.

### 3 Camera Control in Computer Games

Camera systems are increasingly becoming decisive elements to the success of computer games. With the advent of near photo-realistic graphics and the use of powerful and captivating story-driven plots to bring life to games, cameras in games have been neglected. As we have seen in Section 2 film practitioners have characterized standard camera configurations and transitions in terms of a number of cinematographic principles. The use of cinematographic properties and intelligent camera controls have the potential to heavily influence the look and feel of games and give game designers access to the film director visual toolbox.

Camera systems in modern day games can be classified into three different categories;

1. *First person camera systems*: users control the camera (allowing them to have to feel of being the character in virtual environment and looking from their eyes). There are a multitude of games that use first person camera views, notably the Doom series.
2. *Third person camera systems*: the camera system relies on tracking the character from a set of fixed positions, constantly changing the cameras posi-

tion based on the environment and the users interactions with the environment. This mode of camera systems presents a problem when the views presented by the camera do not co-adhere with the current events in the game, *e.g.* when a character leans on to a wall, the camera defaults to moving to the front of the player, disrupting the game play by blocking the view of the opponents.

3. *Action replay camera systems*: replays are heavily used in modern games, to highlight notable scenes in a certain game, and it is imperative that the images generated by the camera system during the replay are meaningful.

For example, *Tomb Raider*<sup>1</sup> is a successful series of computer games that has been both widely praised and criticized for its use of a dynamic camera. The game utilizes a third-person view with respect to the main character, where the camera is *attached* to the character. The camera system employed was rather limited in expressing informative shots when in tight spots; often leading to situations where the camera displayed awkward views, preventing the user from playing the game as it was intended. The camera system computed its next best position without significant consideration of the visual properties and the ongoing action within the environment.

*Full Spectrum Warrior*<sup>2</sup>, an action war military simulator, uses an advanced (in computer games terms) camera system to allow the player to effectively manage teams of soldiers. Its main feature is the *auto look* facility which helps the user by presenting a shot that handles occlusion to prevent the view from being blocked by an object (wall) through the use of ray casting. The fly-by sequences performed by the camera also avoid collisions with environmental objects by applying the same occlusion detection method. *Jump cuts* are used to handle situations when the only evident path is to move through a wall or an obstacle, whereby the camera jumps to the scene beyond the wall, avoiding the unnatural view of going through a wall. While *Full Spectrum Warrior* does constitute a step forward in the use of cameras in games, it lacks the cinematic expression that is apparent in film. Cinematic camera transitions, which preserve the flow of a story, are not apparent in camera systems games as they require heavy computations to handle positioning, orientation, occlusion handling and other image properties as well as defining good shots based on a certain narrative.

Camera shots are the heart of producing truly interactive visual applications. The consideration of cinematic properties and literature (narrative) should provide cues to automatic generation of camera shots that are both intelligent and meaningful, enabling games that give the look and feel of film. Games present an inherent difference to film as most games are controlled by the player and define a dynamic environment. Therefore the problem for automated camera systems for games is intrinsically more complex than for their static counterparts, or for systems where the temporal evolution of action is known in advance. While automation of camera shots based on cinematographic principles present meaningful shots, the use of editing techniques (which are very rare indeed within

<sup>1</sup> Tomb Raider, Eidos plc, [www.eidos.com](http://www.eidos.com)

<sup>2</sup> Full Spectrum Warrior, Pandemic Studios, [www.pandemicstudios.com](http://www.pandemicstudios.com)

games today) can preserve the game-play by presenting jump-shots or cut-scenes to show the user only what is intended. Indeed, such technologies would reduce the confusion that is evident in many games.

## 4 Virtual Camera Planning Systems

In this section, we present a review of a number of approaches to assist or automate the setting up of a virtual camera. The problem is a complex one involving the geometric properties of 3D model worlds together with the higher level problem of describing the desired properties of the output. Our review is structured around two criteria: the expressivity of the properties and the nature of the solving mechanisms.

### 4.1 Expressivity

Most systems share a similarly declarative approach and identify a common set of properties by which a shot may be specified. We distinguish *on-camera* and *on-screen* properties. On the one hand, *on-camera* properties define the location, orientation and movement of the camera in the 3D scene. This encompasses scale shots (*e.g. establishing shot, long shot, close shot*) that define the distance from the camera to the main subject, the vantage point (*e.g. low angle, high angle*), and classical camera motions such as *travelling, dolly*, and *zoom* shots. Most recent static camera planning [14,25,19,28] and dynamic camera planning approaches [32,12,4,18,20,11] rely on this grammar by providing a description language. Pickering *et al.* [29] propose an IDL language (Image Description Language) that meets these requirements.

By contrast, *on-screen* properties allow the specification of where the objects are located in the screen (in a relative, absolute or approximate manner) and other properties of the shot itself (*e.g. size, orientation of objects and possible occlusion*). Here the approaches differ in the range of properties they consider and in their qualitative or quantitative nature. For example, Blinn's initial work [7] and its derivatives [8,10] abstract objects as points and their exact location on the screen must be provided. Such accounts are intrinsically restricted to no more than two objects on the screen otherwise the specification becomes over-constrained. Some interactive approaches [17] offer a precise control of characteristic points (*e.g. edges of a cube*) by controlling the path of their image on the screen. In a less contrived way, some approaches allow the specification of object position with respect to on-screen areas such as rectangular frames [20,11,3,28]. These match the level of specification found in storyboards, but can prove inadequate for complex objects.

The occlusion property (*i.e. the object to view should not be, or should only be partially, occluded*) is fundamental in camera planning. Most approaches use abstractions of the objects, such as spheres, and compute occlusion using their projection on the screen [25,11]. Though easy to compute, it is clear that



for many objects and scenes sphere-based approximation of objects cannot adequately model occlusion. Bares *et al.* [6] prune the search space by projecting the occluders onto discretised hemispheres located in the center of each object and by intersecting all the volumes. More sophisticated techniques that rely on ray-casting [23] and hardware rendering capacities [12,18] provide effective occlusion tests.

While a broad range of properties has been implemented, many compositional techniques have been neglected such as perspective lines, simple primitives and virtual curves shaped by a set of objects of interest in the screen.

## 4.2 Solving Mechanisms

We propose a description of solving mechanisms based on four classes of approaches:

- algebraic systems** represent the problem in vector algebra and directly compute a solution;
- interactive systems** set up cameras in response to directives from a user, who continuously observes the output image;
- reactive real-time systems** rely on robot motion planning mechanisms in a dynamic virtual environment;
- optimisation and constraint-based systems** model the properties as constraints and objective functions and rely on a broad range of solving processes that differ in their properties (*e.g.* completeness, incompleteness, and softness).

**Algebraic Systems.** In an algebraic system a camera set-up is regarded as the solution to a vector algebra problem defined on the model world being viewed. This limits the systems to examples of this class, imposing a lack of flexibility on the system. The congruent advantage being that the dedicated solution is usually computationally efficient. The assumptions made on the graphical entities are strong as objects are to be considered as points and the expressivity is limited to locating two entities in the screen space.

The earliest example of such a system was the work of Blinn [7] at NASA. Working on the visualization of space probes passing planets, he developed a system for setting up a camera so that the probe and planet appeared on-screen at given coordinates with given sizes. Attempts to generalize these systems have relied on the use of idioms, standard lay-outs of subjects and cameras commonly used in cinematography. In these systems solution methods are devised for each lay-out, and the input consists of a model world together with a list of idioms to apply. Such system have been developed by Butz [8] and Christianson [10].

Vector algebra approaches have also been studied in purely 2D-based applications such as cel animation (motion-picture cartoons) or virtual 2D guided tours (presentation of different artworks such as frescos, tapestries or paintings). The need for camera planning algorithms for cartoon animation has been tackled by Wood *et al.* in [32], in association with the Walt Disney Animation Studios.

Their system generates the panoramas by relying on some basic camera moves such as pan, tilt-pan, zoom and truck. Two dimensional “multimedia guided tours” aim is to help the visitors during their visit by providing additional information on the artworks. Zancanaro *et al.* explored the opportunities to use the PDAs in multimedia museum tours in [34,33]. Their approach consists of planning camera movements on *still* images *i.e.* the frescos or paintings exposed in the museum. In a similar way, Palamidese [26] proposes the description of art by planning camera movements that first show details and then zooms out to show an entire work.

**Interactive Systems.** Interactive control systems provide the user with a view of the model world and modify the camera set-up in response to input by the user. The immediate problem with such systems is how a user’s input devices will map onto the properties of the camera. Ware and Osborne [31] reviewed the possible mappings, which they referred to as camera control metaphors. One of the first systems to implement such control was developed by Phillips [27], for the human figure modelling system *Jack*. *Jack* could not properly manipulate model figures about axes parallel or perpendicular to the view direction. The camera control system prevented this occurring by repositioning the camera. It could also reposition the camera to make a selected object visible if it was off screen and manage occlusion via z-buffer rendering.

A system that allows a user to move a camera through a dynamic environment using a pointing device, was developed by Gleicher and Witkin [17]. In this approach the difference between the actual screen properties and the desired properties input by the user was treated as a force. This was applied to the camera set-up, which itself was treated as a body in a classical mechanics system.

**Reactive Real-Time Approaches.** Applications of camera control to reactive environments require specific approaches that emphasises on continuity, smoothness and occlusion criteria w.r.t. a target. Virtual camera planning while following a single target is very close to visual servoing in robotics (specifying a task as the regulation in the image of a set of visual features), and can share similar solving techniques. In [12], a visual servoing approach is proposed that integrates constraints on the camera trajectory. If the primary task (following the object) does not instantiate all the camera parameters, secondary tasks may be added (*e.g.* occlusion or lighting).

A similar application in the domain of computer games has been explored by Halper *et al.* [18]. Here an ad-hoc incremental solving process has been proposed to satisfy at each frame a set of constraints on the screen (*e.g.* height angle, angle of interest, size, visibility).

**Optimisation and Constraint-Based Approaches.** Static virtual camera planning also referred as virtual camera composition (VCC) and dynamic virtual camera planning (VCP) problems can both be viewed as constrained and/or optimisation problems. The set of properties of the shots (*e.g.* framing, orientation, zoom factor) can be expressed as numerical constraints (properties that

must hold) or objective functions (properties to be maximized/minimized) on the camera variables (respectively the camera’s path variables). The general process is to search the space of camera set-ups for one that maximizes/minimizes the objective functions while satisfying the constraints. A broad range of procedures is then available and the solvers differ in the way to manage over-constrained and under-constrained cases, in their complete or incomplete search capacities, in local minima failures and possible optimisation processes (generally finding the best solution with respect to an objective function).

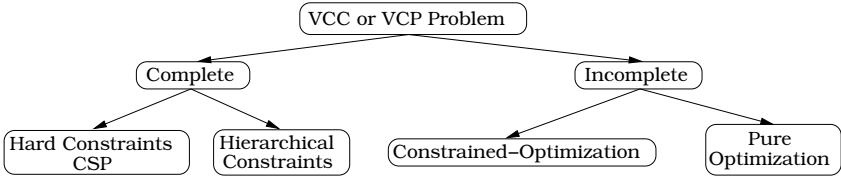
We classify the “constraint-and-optimisation”-based approaches the following way:

**complete methods** perform an exhaustive exploration of the search space, thus providing the user with the whole set of solutions to the problem, or none if the problem is inconsistent. This is generally achieved by a computational and time expensive dynamic programming approach (*split* and *explore*).

**incomplete methods** are mostly based on a stochastic investigation of the search space, thereby computing a unique solution. The output of an incomplete method can either be a solution to the problem, or an approximation of a solution as these methods try to minimize errors given objective functions.

**hybrid methods** rely on cooperation between different techniques to simultaneously manage constraints and objective functions.

The classification of the solving strategies for the VCC and VCP problems is summarized in Figure 1.



**Fig. 1.** Taxonomy of the optimisation and constraint-based approaches to VCC (Virtual Camera Composition) and VCP (Virtual Camera Planning) problems

*Complete Approaches.* The Constraint Satisfaction Problem (*CSP*) framework has proven to succeed in some camera composition and motion planning approaches. However these can differ in the way they handle over-constrained problems. We distinguish the *hard constraints* approaches [20,11] from the *hierarchical constraints* ones [4]. Hard constraint approaches such as Jardillier & Langu  nou [20] (the *Virtual Cameraman*) use pure interval methods to compute camera paths which yield sequences of images fulfilling temporally indexed image properties. The use of pure interval methods has been improved by Christie *et al.* in [11] by describing camera trajectories as sequences of parameterized elementary camera movements called *hypertubes*. Unlike most approaches which

only guarantee the correctness of user-defined properties for a set of points on a camera trajectory (generally the starting and ending points plus some keypoints taken on the camera path), interval-based methods guarantee the fulfillment of the properties during a whole film sequence. Unfortunately, the benefits of completeness of interval-based techniques are counterbalanced by the computational effort required, and the fact that there is no mechanism for constraint relaxation. However, since the method explores the whole search space, the user has a guarantee that there are no solutions whenever the method fails.

In contrast, hierarchical constraints approaches are able to relax some of the constraints in order to give the user an approximate solution of the problem. Bares *et al.* propose to use a partial constraint satisfaction system named CONSTRAINTCAM [4] in order to provide alternate solutions when constraints cannot be completely satisfied. If CONSTRAINTCAM fails to satisfy all the constraints of the original problem, the system relaxes weak constraints and, if necessary, decompose a single shot problem to create a set of camera placements that can be composed in multiple viewports [5] providing an alternate solution to the user.

*Incomplete Approaches.* In general, incomplete approaches allow more flexibility in the definition of the problem; the main difference is in the kind of solving procedure applied to the problem. The incomplete search can be based on pure *optimisation* procedures (*e.g.* descent methods) or stochastic search methods (*e.g.* local search, genetic algorithms, simulated annealing).

Indeed as early as 1992, Drucker *et al.* [13] proposed CINEMA, a general system for camera movement. CINEMA was designed to address the problem of combining different paradigms (*e.g.* *eyeball in hand*, *scene in hand*, or *flying vehicle* metaphors [31]) for controlling camera movements. This early approach incited the authors to explore the constraint satisfaction methodology to address some of the CINEMA's problems. In [14,15] Drucker and Zeltzer improved their previous ideas and developed the CAMDROID system which specifies behaviours for virtual cameras in terms of task level goals (objective functions) and constraints on the camera parameters. They regroup some primitives constraints into *camera modules* which represent a higher level means of interaction with the user. The constraints of each module are then combined by a constraint solver and solved by a camera optimizer based on the *CFSQP* (Feasible Sequential Quadratic Programming coded in C) package, which has been designed to solve large scale constrained non-linear optimisation problems [14].

Unfortunately this initial attempt at constrained optimisation effort does not offer a systematic solution for handling constraint failures that can occur frequently in dynamic environments with complex scene geometries and requires a good initial guess for correct convergence.

In order to address the major drawbacks of their CONSTRAINTCAM system, Bares *et al.* [3,2] proposed the use of a heuristic search method that uses a constraint's allowable minimum and maximum values to reduce the size of the 7 degrees of freedom search space of possible camera positions ( $x, y, z$ ), orientations ( $dx, dy, dz$ ) and field of view angles (fov). Their method combine both constraints

and objective functions in a constrained-optimisation algorithm. This was addressed by Pickering and Olivier [29], who defined an Image Description Language (IDL), a context free grammar allowing properties of images to be defined. The IDL could be parsed into hierarchies of constraints and objectives, which were then subject to constrained optimisation using a genetic algorithm [28].

However, optimisation-based techniques rise the delicate question of how to provide the weights of each property and how to aggregate them in the objective function in a way that is stable and efficient whatever the description may be. Moreover, most incomplete techniques require a fine and specific tuning of the solving process (parameters of the simulated annealing, cross-over probabilities in GA, and stopping condition).

*Hybrid Approaches.* Some parallel works have proposed hybrid approaches through a cooperation of different methods. The novel feature is that a first step computes volumes solely defined on camera positions to build models of the feasible space [3,28]. And a second step relies on a classical stochastic search as in [28] or heuristic search [3]. This provides an efficient pruning of the search space and retains the search process in *interesting* volumes.

## 5 Discussion

In order to manage complex 3D scenes, an abstraction of the geometry is necessary: objects are mostly considered as simple primitives (points or bounding volumes such as spheres) and provide imprecise and possibly erroneous results since most complex objects can not be represented with simple bounding volumes. Some accounts do consider the precise geometry for occlusion purposes [18,28], but due to the computational cost of this process, have to rely on hardware rendering capacities. Improving the quality of abstraction of the objects is a difficult but necessary work that requires proposing an adequate model as well as effective solving mechanisms. Complex situations, such as filming a character through the leaves of a tree, have not yet been addressed.

The expressiveness of the set of properties is mostly related to the application context and the retained solution mechanism. Algebraic, interactive and real-time approaches generally rely on quantitative relations (*e.g.* exact locations or orientations of objects on the screen) [7,12,18,10,17,3], while optimisation and constraint-based systems allow for qualitative relations through the use of square or oval frames to constrain the location of objects [25,19,20,11]. In the latter, qualitative relations allow to relax the hardness of the properties. Expressiveness is provided through the use of property softness too. In Drucker's work [13], the choice between constraints and objective functions is hidden to the user, but usually one has to fix the softness of the constraints through scalar coefficients [5,3,25] which can be awkward to set. The question remains as to how one decides on the weights of each property and the aggregation of the weights in a single objective function. Hard constraint-based approaches do not require any user settings [20,11] and uses constraint relaxation to compute approximate solutions [4].

The solution technique adopted constrains both the geometry abstraction and expressiveness of the approaches. However, compared to algebraic and real-time approaches, constraint and optimisation-based approaches provide a powerful framework to add new constraints or objective functions relative to any specific property. The problem is the computational cost of these mechanisms, although hybrid approaches provide some valuable results as efficient search techniques can be applied in promising areas of the search space.

Long-term research is required into higher levels notions such as perceptual and aesthetic properties. To some extent this requires an adequate cognitive model in order to assist the user in his mental time and space construction of the world. Indeed, some work has adopted common set of editing rules to effectively engage the user [16,30]. However, the editing choices rely primarily on the nature of actions in the environment rather than on an emotional state of the user. Furthermore, the incorporation of aesthetic properties requires an important collaboration with cognitive psychology and relies on empirical characterization of the nature of composition.

## References

1. D. Arijon. *Grammar of the Film Language*. Hastings House Publishers, 1976.
2. W. Bares and B. Kim. Generating Virtual Camera Compositions. In *IUI '01: Proceedings of the 6th international conference on Intelligent user interfaces*, pages 9–12, New York, NY, USA, 2001. ACM Press.
3. W. Bares, S. McDermott, C. Boudreaux, and S. Thainimit. Virtual 3D Camera Composition from Frame Constraints. In *MULTIMEDIA '00: Procs. of the eighth ACM international conference on Multimedia*, pages 177–186. ACM Press, 2000.
4. W. H. Bares, J. P. Gregoire, and J. C. Lester. Realtime Constraint-Based Cinematography for Complex Interactive 3D Worlds. In *Procs of AAAI-98/IAAI-98*, pages 1101–1106, 1998.
5. W. H. Bares and J. C. Lester. Intelligent Multi-Shot Visualization Interfaces for Dynamic 3D Worlds. In *IUI '99: Proceedings of the 4th international conference on Intelligent user interfaces*, pages 119–126, New York, NY, USA, 1999. ACM Press.
6. W. H. Bares, D. W. Rodriguez, L. S. Zettlemoyer, and J. C. Lester. Task-sensitive cinematography interfaces for interactive 3d learning environments. In *Proceedings Fourth International conference on Intelligent User Interfaces*, pages 81–88, 1998.
7. J. Blinn. Where am I? what am I looking at? *IEEE Computer Graphics and Applications*, pages 76–81, July 1988.
8. A. Butz. Animation with CATHI. In *Proceedings of American Association for Artificial Intelligence/IAAI '97*, pages 957–962. AAAI Press, 1997.
9. A. Butz, A. Krüger, and P. Olivier, editors. *Smart Graphics, Third International Symposium, SG 2003, Heidelberg, Germany, July 2-4, 2003, Proceedings*, volume 2733 of *Lecture Notes in Computer Science*. Springer, 2003.
10. D. B. Christianson, S. E. Anderson, L. He, D. H. Salesin, D. S. Weld, and M. F. Cohen. Declarative Camera Control for Automatic Cinematography. In *Proceedings of the American Association for Artificial Intelligence 1996*, pages 148–155, 1996.

11. M. Christie and E. Langu  nou. A Constraint-Based Approach to Camera Path Planning. In Butz et al. [9], pages 172–181.
12. N. Courty and E. Marchand. Computer animation: A new application for image-based visual servoing. In *Proceedings of IEEE Int. Conf. on Robotics and Automation, ICRA'2001*, volume 1, pages 223–228, 2001.
13. S. M. Drucker, T. A. Galyean, and D. Zeltzer. Cinema: A System for Procedural Camera Movements. In *SI3D '92: Proceedings of the 1992 symposium on Interactive 3D graphics*, pages 67–70, New York, NY, USA, 1992. ACM Press.
14. S. M. Drucker and D. Zeltzer. Intelligent Camera Control in a Virtual Environment. In *Proceedings of Graphics Interface '94*, pages 190–199, Banff, Alberta, Canada, 1994.
15. S. M. Drucker and D. Zeltzer. Camdroid: A System for Implementing Intelligent Camera Control. In *Symposium on Interactive 3D Graphics*, pages 139–144, 1995.
16. D. A. Friedman and Y. A. Feldman. Knowledge-based cinematography and its applications. In *Proceedings of the 16th European Conference on Artificial Intelligence, ECAI'2004*, pages 256–262. IOS Press, 2004.
17. M. Gleicher and A. Witkin. Through-the-lens camera control. In *Proceedings of ACM SIGGRAPH'92*, pages 331–340, 1992.
18. N. Halper, R. Helbing, and T. Strothotte. A camera engine for computer games: Managing the trade-off between constraint satisfaction and frame coherence. In *Proceedings of the Eurographics'2001 Conference*, volume 20, pages 174–183, 2001.
19. N. Halper and P. Olivier. CAMPLAN: A Camera Planning Agent. In *Smart Graphics 2000 AAAI Spring Symposium*, pages 92–100, March 2000.
20. F. Jardillier and E. Langu  nou. Screen-Space Constraints for Camera Movements: the Virtual Cameraman. In N. Ferreira and M. G  bel, editors, *Procs. of EUROGRAPHICS-98*, volume 17, pages 175–186. Blackwell Publishers, 1998. ISSN 1067-7055.
21. S. Katz. *Film Directing Shot by Shot: Visualizing from Concept to Screen*. Michael Wiese Productions, 1991.
22. J. Mascelli. *The Five C's of Cinematography: Motion Picture Filming Techniques*. Cine/Grafic Publications, Hollywood, 1965.
23. S. McDermott, J. Li, and W. Bares. Storyboard Frame Editing for Cinematic Composition. In *IUI '02: Proceedings of the 7th international conference on Intelligent user interfaces*, pages 206–207, New York, NY, USA, 2002. ACM Press.
24. C.F. Nodinem, J.J. Locher, and E.A. Krupinski. *The role of formal art training on perception and aesthetic judgement of art compositions*. Leonardo, 1993.
25. P. Olivier, N. Halper, J. Pickering, and P. Luna. Visual Composition as Optimisation. In *AISB Symposium on AI and Creativity in Entertainment and Visual Art*, pages 22–30, 1999.
26. P. Palamidese. A Camera Motion Metaphor Based on Film Grammar. *Journal of Visualization and Computer Animation*, 7(2):61–78, 1996.
27. C. B. Phillips, N. I. Badler, and J. Granieri. Automatic viewing control for 3d direct manipulation. In *Proceedings of the 1992 symposium on Interactive 3D graphics*, pages 71–74. ACM Press New York, NY, USA, 1992.
28. J. H. Pickering. *Intelligent Camera Planning for Computer Graphics*. PhD thesis, Department of Computer Science, University of York, 2002.
29. J. H. Pickering and P. Olivier. Declarative Camera Planning Roles and Requirements. In *Proceedings of the Third International Symposium on Smart Graphics*, volume 2733 of *Lecture Notes in Computer Science*, pages 182–191. Springer, 2003.

30. B. Tomlinson, B. Blumberg, and D. Nain. Expressive autonomous cinematography for interactive virtual environments. In Carles Sierra, Maria Gini, and Jeffrey S. Rosenschein, editors, *Proceedings of the Fourth International Conference on Autonomous Agents*, pages 317–324, Barcelona, Catalonia, Spain, 2000. ACM Press.
31. C. Ware and S. Osborne. Exploration and virtual camera control in virtual three dimensional environments. In *SI3D '90: Proceedings of the 1990 symposium on Interactive 3D graphics*, pages 175–183, New York, NY, USA, 1990. ACM Press.
32. D. N. Wood, A. Finkelstein, J. F. Hughes, C. E. Thayer, and D. H. Salesin. Multiperspective panoramas for cel animation. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 243–250, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.
33. M. Zancanaro, C. Rocchi, and O. Stock. Automatic video composition. In Butz et al. [9], pages 192–201.
34. M. Zancanaro, O. Stock, and I. Alfaro. Using cinematic techniques in a multimedia museum guide. In *Proceedings of Museums and the Web 2003*, March 2003.



# Graphical Data Displays and Database Queries: Helping Users Select the Right Display for the Task

Beate Grawemeyer and Richard Cox

Representation & Cognition Group,  
Department of Informatics,  
University of Sussex, Falmer, Brighton BN1 9QH, UK  
{beateg, richc}@sussex.ac.uk

**Abstract.** This paper describes the process by which we have constructed an adaptive system for external representation (ER) selection support, designed to enhance users' ER reasoning performance. We describe how our user model has been constructed - it is a Bayesian network with values seeded from data derived from experimental studies. The studies examined the effects of users' background knowledge-of-external representations (KER) upon performance and their preferences for particular information display forms across a range of database query types.

## 1 Introduction

Successful use of external representations (ERs) depends upon skillful matching of a particular representation with the demands of the task. Numerous studies (*eg.* [6] and [13]) have shown how a good fit between a task's demands and particular representations can facilitate search and read-off of information. For example, [18] provides a review of studies that show that tasks involving perceiving relationships in data or making associations are best supported by graphs, whereas 'point value' read-off is better facilitated by tabular representations.

Numerous factors are associated with ER-task matching skill. Firstly, it is known that individuals differ widely in terms of their preferences for particular forms of ERs [3]. Better reasoners organise their knowledge of ERs on a 'deeper' semantic basis than poorer reasoners, and are better at correctly naming various ER forms [4].

Secondly, some types of tasks require a particular, specialised type of representation whereas for other types of tasks, several different ER forms may be useful. The extent to which a problem is representationally-specific is determined by characteristics such as its degree of determinacy (extent to which it is possible to build a single model of the information in the problem).

ER selection skill requires, *inter alia*, knowledge of a range of ERs in terms of a) their semantic properties (*eg.* expressiveness, b) their functional roles together with information about the 'applicability conditions' under which a representation is suitable for use ([17][2][14]).

Considerable advances in intelligent automatic matching of information to visual representations have been made, beginning with APT [11], which included a composition algebra and primitives to generate a wide range of information displays. Later, SAGE [16] extended APT's graphic design capabilities. Another system, BOZ, [1] utilised a task-analytic approach. However, the APT, SAGE and BOZ systems do not accommodate differences between users in terms of their background knowledge of ERs or ER preferences. Individuals differ widely in terms of their ER knowledge and selection predilections. Whilst most people can use some ER forms effectively (*eg.* common examples like bar or pie charts), some ER types require specialised knowledge and training. Euler's circles are examples of the latter kind - set diagram semantics have to be learned specifically [3]. Individuals also differ in terms of their preferences for representing information visually (*eg.* via graphics or diagrams) or verbally (lists, notes, memoranda) [10].

The aim of this paper is to describe the process by which we constructed an adaptive system that recommends ERs taking into account the users' background knowledge-of-external representations (KER) and his/her preferences for particular types of information display.

This paper extends our earlier work (*eg.* [7]) by further researching the relationships between individuals' background knowledge of external representations and their ability to select appropriate information displays. The domain studied was that of diagrammatic displays of database information. The work was conducted using an unintelligent database system (AIVE). The results have informed the design of an adaptive system (I-AIVE) capable of supporting users in their choice of external representations (ERs). I-AIVE's user model is being developed on the basis of empirical data gathered from a series of experimental studies ([7]). This approach is similar to that of [9], who used empirical data in the validation of the READY system. That system models users' performance capacity under various cognitive load conditions.

The structure of this paper as follows: Section 2 covers the experimental procedure used to investigate the effect of users' background knowledge of ERs upon information display selection on different representation-specific database query tasks. The experimental results and implications for the ER recommender system are discussed in chapter 3. Section 4 outlines the user model implementation in the form of a Bayesian network which is 'seeded' with and derived from the empirical data. Part 5 describes the adaptation process, where the user model permits the system to engage in overt adaptive behaviors such as suggesting or providing ER selection hints or covert adaptive behaviours, such as restricting the choice of representations for particular participants for particular tasks in order to encourage good ER-to-task matching behavior. Conclusions are presented in section 6.

## 2 Experiment

In our experiment a prototype automatic information visualization engine (AIVE) was used to present a series of questions about the information in a database.

*Knowledge of External Representations (KER) Tasks.* Twenty participants first completed 4 tasks designed to assess their knowledge of external representations (KER). These consisted of a series of cognitive tasks designed to assess ER knowledge representation at the perceptual, semantic and output levels of the cognitive system [5]. A large corpus of external representations (ERs) was used as stimuli. The corpus contained a varied mix of 112 ER examples including many kinds of chart, graph, diagram, tables, notations, text examples, *etc.*

The first task was a decision task requiring decisions, for each ER in the corpus, about whether it was ‘real’ or ‘fake’<sup>1</sup>. This was followed by a categorisation task designed to assess semantic knowledge. Participants categorised each representation as ‘graph or chart’, ‘icon/logo’, or ‘map’, *etc.* In the third (functional knowledge) task, participants were asked ‘*What is this ER’s function?*’ An example of one of the (12) multiple-choice response options for these items is ‘*Shows patterns and/or relationships of data at a point in time*’. In the final task, participants chose, for each ER in the corpus, a specific name from a list. Examples include ‘venn diagram’, ‘timetable’, ‘scatterplot’, ‘Gantt chart’, ‘entity relation (ER) diagram’. The 4 tasks were designed to assess ER knowledge representation using an approach informed by picture and object recognition and naming research [8]. The cognitive levels ranged from the perceptual level (real/fake decision task) to through production (ER naming) to deeper semantic knowledge (ER functional knowledge task).

*AIVE Database Query Task.* Following the KER tasks, participants underwent a session of 30 trials with the AIVE system. The AIVE database contains information about 10 types of car: manufacturer, model, purchase price, CO<sup>2</sup> emission, engine size, horsepower, *etc.*

On each trial, AIVE (figures 1, 2 and 3) presented a database query (*eg.* ‘Which two cars are most similar with respect to their Co2 emission and cost per month?’). Participants could then choose between various types of ERs, *eg.* set diagram, scatter plot, bar chart, sector graph, pie chart and table (all representations were offered by the system for any query). These options were presented as an array of buttons each with an icon depicting, in stylised form, an ER type<sup>2</sup> (table, scatterplot, pie chart, ...)(see figure 1). Participants were told that they were free to choose any ER, but that they should select a form of display they thought was most likely to be helpful for answering the question. Following the participant’s selection, AIVE displayed a ‘full’ (data instantiated) version of that representation using data from the database of car information - examples are shown in Figures 2 and 3.

Across the 30 trials, participants experienced six types of database query: ‘identify’; ‘correlate’; ‘quantifier-set’; ‘locate’; ‘cluster’ and ‘compare negative’. For example, a typical ‘correlate’ task was: ‘Which of the following statements is

<sup>1</sup> Some items in the corpus are invented or chimeric ERs.

<sup>2</sup> The spatial layout of the representation selection buttons was randomized across the 30 query tasks in order to prevent participants from developing a set pattern of selection.

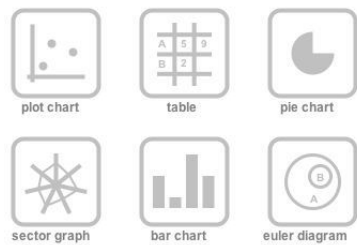


Fig. 1. AIVE representation selection interface

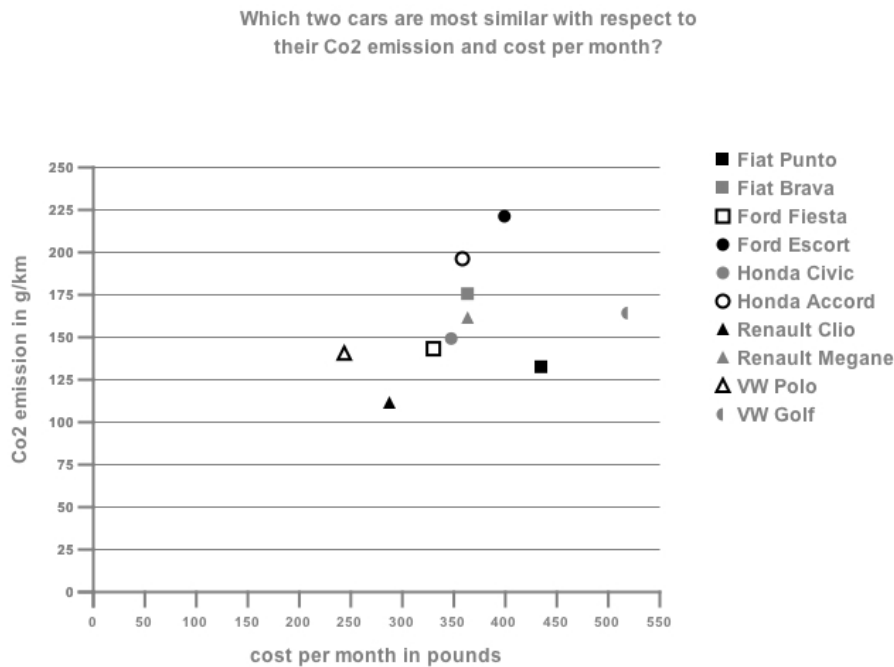
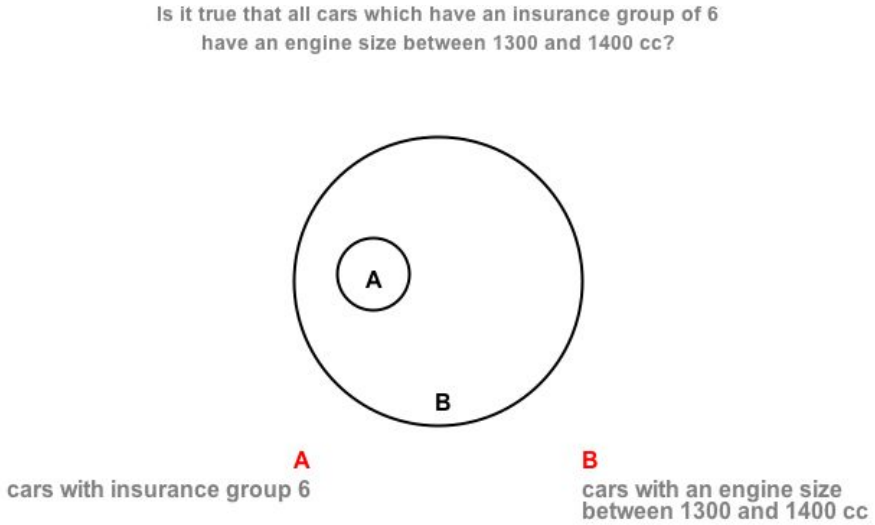


Fig. 2. Example of AIVE plot representation

true? A: Insurance group and engine size increase together. B: Insurance group increases and engine size decreases. C: Neither A nor B?'; or a typical locate task: 'Where would you place a Fiat Panda with an engine size of 1200 cc inside the display?'.

Based on the literature (*eg.* [6]), a single 'optimal' ER for each database query form was identified<sup>3</sup>. However, each AIVE query type could *potentially* be answered with any of the representations offered by the system (except for some 'cluster' tasks for which a set diagrams was the only really usable ER).

<sup>3</sup> Display selection accuracy (DSA) scores were based on this (see results section).



**Fig. 3.** Example of AIVE Euler's circles representation

Participants were not permitted to select a different representation following their initial selection. This constraint was imposed in order to encourage participants to carefully consider which representation was best matched to the task. Following a completed response, participants were presented with the next task and the sequence was repeated. The following data were recorded by the AIVE system: (1) the randomized position of each representation icon from trial to trial; (2) the users' representation choices; (3) time to read question and select representation (selection); (4) time to answer the question using chosen representation (answer); and (5) participants' responses to questions.

### 3 Results and Discussion

To recapitulate there were 20 participants, each of whom was presented with 30 AIVE tasks (600 data points in total). The independent and dependent variables are shown in Table 1. Statistical analyses indicate that the KER tasks are significant predictors of display selection accuracy (DSA) and database query answer (DBQA) performance.

DSA was significantly predicted by one of the KER tasks (ER classification knowledge). For DBQA, the best KER predictor was ER functional knowledge. Hence a degree of conceptual (classificatory) knowledge of ERs predicts success at appropriate information display selection on the AIVE tasks but deeper semantic (functional) knowledge of ERs is associated with success at *using* the selected ER *ie.* reading-off information and using it to respond correctly to the database query. Additionally, appropriate representation selection results in better query performances. This suggests that, for predicting query response accu-

racy, a participant's KER can be as powerful a predictor of question answering accuracy as display selection accuracy.

The selection latency results show that a speedy selection of a display type in AIVE is associated with a good display-type choice. This implies that users either recognise the 'right' representation and proceed with the task or they procrastinate and hesitate because of uncertainty about which display form to choose. Less time spent responding to the database query question is associated with a good display-type choice and correct query response. This suggests that the selection and database query latencies may be used in the system's user model as predictors of users' ER expertise.

The results reported so far were based on all the AIVE query types combined. However, they differed extensively in terms of their representational specificity. Two different query types were contrasted in order to examine the effects of the tasks' 'representational specificity'.

### 3.1 Comparing a Highly ER-Specific AIVE Task and a Less ER-Specific AIVE Task

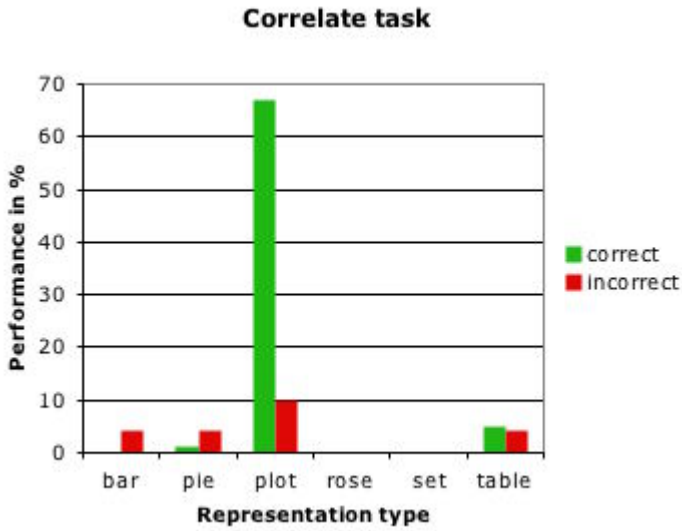
Participants' selection behavior and database query task performance for the correlate and locate task are shown in figure 4 and 5. The correlate task is highly representation specific and the locate task much less so.

**The AIVE Correlate Task - High ER-Specificity.** As shown in figure 4, 77% of AIVE correlate type queries were answered correctly by participants. Moreover, in 77% of the cases they chose the most appropriate ER display (scatter plot) from the array of display types (ERs) offered by AIVE.

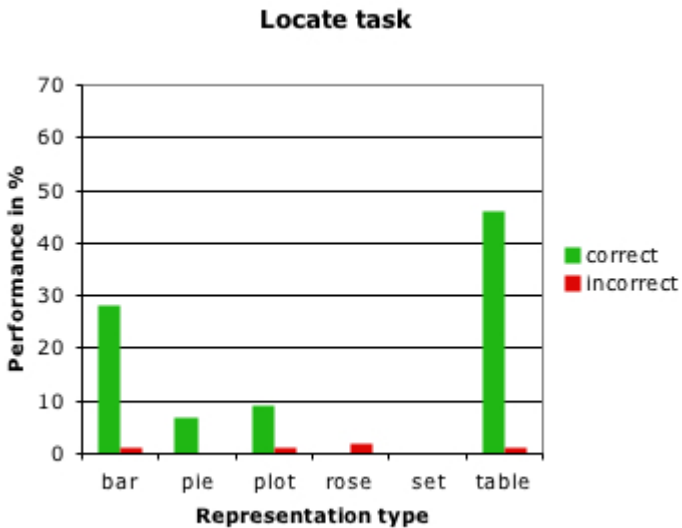
Statistical analysis shows that performance on two of the KER tasks (ER classification and functional knowledge) predicts good display selection performance (see Figure 6). Especially ER classification knowledge of set diagrams, and functional knowledge of graphs and charts (as might be expected). Longer display selection latency is associated with longer time spent responding to the database query question.

**The AIVE Locate Task - Low ER-Specificity.** Figure 5 shows that 4 different data displays are effective for this task. Overall, participants locate task queries were answered with a high degree of accuracy (94%). However, in only 51% cases did participants choose the 'right' representation (table or matrix ER). A range of other AIVE display forms were also effective (bar and pie charts, scatterplots).

KER and database query answer performance were not significantly correlated (Figure 7), which implies that less 'graphical literacy' on the part of participants is required for this task compared to the correlate task. Database query answer performance and display selection accuracy were not significantly correlated - as would be expected on a task in which accurate responding to database queries can be achieved by using any one of 4 different information displays.



**Fig. 4.** The highly representation-specific ‘correlate’ task. DBQA performance in % as a function of chosen representation type.



**Fig. 5.** The less representation-specific ‘locate’ task. DBQA performance in % as a function of chosen representation type.

## 4 User Model Implementation

The experimental results show that particular types of data are crucial for modeling. Machine learning techniques vary in terms of their advantages and disadvantages for particular applications and domains, as well as for the underlying information or user data needed for the adaptation process ([12]). Our user model needs to reflect the relationship of KER and the varied degrees of representational specificity of the database query tasks. It also needs to track and predict participants' selection accuracy and database query answering performance for various display and response accuracy relationships within and across the various database query task types. The system should be capable of being more stringent in its recommendations to users on highly representationally-specific task types such as correlate tasks but could be able to be more lenient on more display-heterogeneous tasks.

A Bayesian network approach (*eg.* [15]) was chosen as a basis for I-AIVE's user model, because such networks are suitable, *inter alia*, for recognizing and responding to individual users, and they can adapt to temporal changes.

Table 1 shows the independent and dependent variables, that were empirically observed.

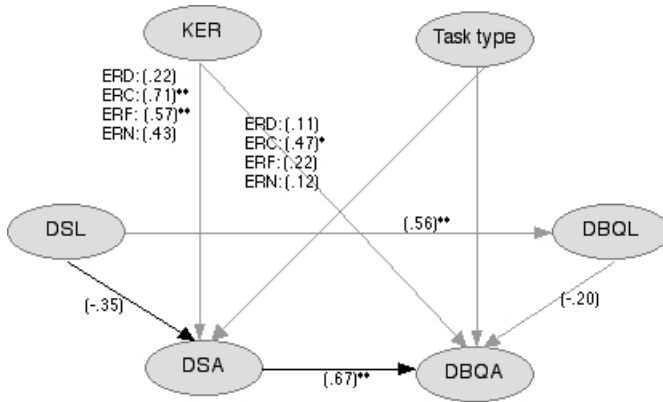
**Table 1.** Independent and dependent variables used for the specification of the network

Independent	
<i>Name</i>	<i>Description</i>
KER	This variable represents users' background knowledge of ERs, gathered in the ER tasks.
Task type	Identify, correlate, quantifier-set, locate, cluster or compare negative.
Dependent	
<i>Name</i>	<i>Description</i>
DSA	This variables covers users' display selection accuracy score. The value of the variable increases, if the user's selected ER is an appropriate ER for the given task. The variables decreases otherwise.
DBQA	Represents the total score of users' response to the database query question. It increases, if the response with the chosen ER was correct. If an incorrect response is given the variable decreases.
DSL	Time to select a representation in milliseconds.
DBQL	Time to answer the question on each trial in milliseconds.

The structure of a simple Bayesian network based on the experimental data can be seen in figures 6 and 7. The correlations between the independent and dependent variables are represented in this network. Figure 6 presents a graph for the highly representation-specific 'correlate' task, and figure 7 shows a graph for the less representation-specific 'locate' task.

The structure of the network represents the relationships between the independent/dependent variables. For example the arc between DSA and DBQL





**Fig. 6.** Graph of a Bayesian network for the highly representation-specific ‘correlate’ task. Correlations between the KER tasks (ERN: naming, ERD: decision, ERC: categorisation and ERF: functional knowledge) and with the AIVE *correlate* task are shown in brackets. \* = correlation is significant at the 0.05 level. \*\* = correlation is significant at the 0.01 level.

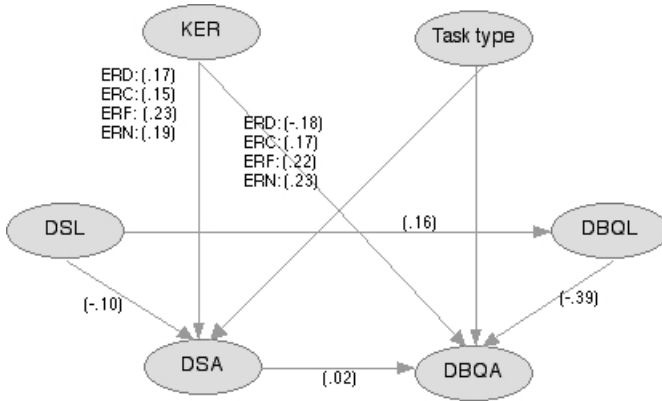
represents the association that good display selection results in better query performance, or the link between DSL and DSA represents that a speedy selection of a display type in AIVE is associated with a good display-type choice. The Bayesian network in I-AIVE’s user model has been ‘seeded’ with the empirical data so that it can monitor and predict users’ ER selection preference patterns within and across query types, relate query response accuracy and latencies to particular display selections and contrive query/display option combinations to ‘probe’ an individual users’ degree of ‘graphical literacy’. The empirical data is to instantiate values in the relevant conditional probability tables (CPTs) at each node of the model. The network will then dynamically adjust the CPT values and evolve individualised models for each of its users as they interact with the system. For example, for each ER selection and resulting database query performance score the corresponding CPT values will be updated and used from the system for an individual adaptation. The learned network is able to make the following inferences:

- **Predicting ER preferences and performance with uncertainty about background knowledge**

If there is uncertainty about users’ background knowledge of ERs, the system is able to make predictions about the dependent variables, through a probability distribution of each these variables.

- **Learning about users’ ER preferences and performance**

Users’ ER preferences and performance can be learned incrementally, through users’ interaction with the system. The network can be updated with the individual characteristics and used to predict future actions and system decisions.



**Fig. 7.** Graph of a Bayesian network for the less representation-specific ‘locate’ task. Correlations between the KER tasks (ERN: naming, ERD: decision, ERC: categorisation and ERF: functional knowledge) and with the AIVE *locate* task are shown in brackets.

These inferences are used as a basis for our system to recommend ERs based on background knowledge, task type and ER preferences.

## 5 The Adaptation Process

I-AIVE’s interventions consist of overt hints or advice to users and also covert adaptations such as not offering less-appropriate display forms in order to prevent users from selecting them. The system is able to adapt to the individual user in the following ways:

- **‘Hiding’ inappropriate display forms**

The system varies the range of ‘permitted’ displays as a function of each task’s ER-specificity and the user’s ER selection skill.

- **Recommending ERs**

The system will interrupt and highlight the most appropriate ER (based on the user model) if too much time is spent on selecting a representation, after learning an individual’s selection display selection latency patterns.

Based on users’ interactions the system will adapt the range of displays and/or recommend ERs. For example, if a user manifests a particularly high error rate for particular task/ER combinations, then the system will limit the ER selection choice and exclude the ER with which the user has problems to answer the particular database query task in the past. After users’ selection display latency for particular task types has been detected, the system is able to recommend ERs if the user is unclear what kind of ER to choose and spends too much time in selection a representation.

## 6 Conclusion and Future Work

In this paper we described our process of how we constructed an adaptive system for external representation selection support, based on experimental data. The aim of the system is to enhance users' ER reasoning performance across a range of different types of database query tasks.

At early stages of user-system interaction, the system only offers display options that it believes lie within the users' 'representational repertoire'. After more extensive user-system interactions the user model will be updated and the system will be able to make firmer recommendations to its user.

The next step in our research will be the evaluation of I-AIVE by comparing two versions in a controlled experiment - one version with the adaptive system turned on and the other version with the user modeling subsystem turned off. The results will be used to inform the development and refinement of the user model.

## References

1. Casner, A.M.: A task-analytic approach to the automated design of information graphics. PhD thesis, University of Pittsburgh (1990)
2. Cheng, P.C.-H.: Functional roles for the cognitive analysis of diagrams in problem solving. In: Cottrell, G.W. (eds.): *Proceedings of the 18th Annual Conference of the Cognitive Science Society*. Mahwah NJ, Lawrence Erlbaum Associates (1996) 207-212
3. Cox, R.: Representation construction, externalised cognition and individual differences. *Learning and Instruction* **9** (1999) 343-363
4. Cox, R., Grawemeyer, B.: The mental organisation of external representations. *European Cognitive Science Conference (EuroCogSci)*, Osnabrück (2003)
5. Cox, R., Romero, P., du Boulay, B., Lutz, R.: A cognitive processing perspective on student programmers' 'graphicacy'. In: Blackwell, A., Marriott, K., Shimojima, A. (eds.): *Diagrammatic Representation & Inference. Lecture Notes in Artificial Intelligence*, Vol. 2980. Springer-Verlag, Berlin Heidelberg (2004) 344-346
6. Day, R.: Alternative representations. In: Bower, G. (eds.): *The Psychology of Learning and Motivation* **22** (1988) 261-305
7. Grawemeyer, B., Cox, R.: A Bayesian approach to modelling user's information display preferences. In: Ardissono, L., Brna, P., Mitrovic, T. (eds.): *UM 2005: The Proceeding of the Tenth International Conference on User Modeling. Lecture Notes in Artificial Intelligence*, Vol. 3538. Springer-Verlag, Berlin Heidelberg (2005) 233-238
8. Humphreys, G.W., Riddoch, M.J.: *Visual object processing: A cognitive neuropsychological approach*. Lawrence Erlbaum Associates, Hillsdale NJ (1987)
9. Jameson, A., Gromann-Hutter, B., March L., Rummer, R.: Creating an empirical basis for adaptation decisions. In: Lieberman, H. (eds.): *IUI 2000: International Conference on Intelligent User Interfaces*. (2000)
10. Kirby, J.R., Moore, P.J., Schofield, N.J.: Verbal and visual learning styles. *Contemporary educational psychology* **13** (1988) 169-184
11. Mackinlay, J.D.: Automating the design of graphical representations of relational information. *ACM Transactions on Graphics* **5(2)** (1986) 110 - 141

12. Mitchell, T.M. (eds.): Machine learning. McGraw Hill, New York (1997)
13. Norman, D.A. (eds.): Things that make us smart. Addison-Wesley, MA (1993)
14. Novick, L.R., Hurley, S.M., Francis, M.: Evidence for abstract, schematic knowledge of three spatial diagram representations. *Memory & Cognition* **27(2)** (1999) 288-308
15. Pearl, J. (eds.): Probabilistic reasoning in intelligent systems: Networks of Plausible Inference. Morgan Kaufmann (1988)
16. Roth, S., Mattis, J.: Interactive graphic design using automatic presentation knowledge. *Human-Factors in Computing Systems* (1994) 112-117
17. Stenning, K. , Cox, R., Oberlander, J.: Contrasting the cognitive effects of graphical and sentential logic teaching: Reasoning, representation and individual differences. *Language and Cognitive Processes* **10(3/4)** (1995), 333 - 354
18. Vessey, I.: Cognitive fit: A theory-based analysis of the graphs versus tables literature. *Decision Sciences* **22** (1991) 219-241

# Visualization Tree, Multiple Linked Analytical Decisions

José F. Rodrigues Jr., Agma J.M. Traina, and Caetano Traina Jr.

University of São Paulo at São Carlos, Av. Trabalhador São-carlense, 400 - Centro, Brazil  
{junio, agma, caetano}@icmc.usp.br

**Abstract.** In this paper we tackle the main problem presented by the majority of Information Visualization techniques, that is, the limited number of data items that can be visualized simultaneously. Our approach proposes an innovative and interactive systematization that can augment the potential for data presentation by utilizing multiple views. These multiple presentation views are kept linked according to the analytical decisions took by the user and are tracked in a tree-like structure. Our emphasis is on developing an intuitive yet powerful system that helps the user to browse the information and to make decisions based both on overview and on detailed perspectives of the data under analysis. The visualization tree keeps track of the interactive actions taken by the user without losing context.

## 1 Introduction and Related Work

The literature of InfoVis presents many visualization techniques [1] proposed to provide means to explore large multidimensional data sets. They comprehend capacities for identifying trends, outliers, correlations, clusters; they can be used to validate and formulate hypothesis, besides demonstrating interesting properties of the data. These visualization techniques have extended the usability of stored data of various natures providing crucial support for decision-making.

However, although the many progresses achieved in this field, there are still unsolved challenges for InfoVis. Existing Information Visualization techniques are usually limited to displaying only around a thousand items [2]. This limitation is innate to visual applications since they rely on human visual perception, which is naturally limited in space. Besides, the task of creating visualization scenes adequate to the human sensorial system is not simple, because the human perception can handle only a limited amount of stimuli, and can be overloaded by messy imagery.

Indeed, according to [3], conventional multivariate visualization techniques do not scale well with respect to the number of objects in the data set, resulting in a display with an unacceptable level of clutter. In [4], for example, it is affirmed that the maximum number of elements that the Parallel Coordinates technique [11] can present, in order to allow the data analysis, is around one thousand. In fact, most of the visualization techniques do not handle much more than that number, generating scenes with a reduced number of just noticeable differences.

To bypass the drawbacks just discussed above, interaction mechanisms emerged presenting a notable synergy with visualization techniques. Therefore, they are deeply concerned concepts in the development of visual exploration software. The essence of

interaction is to provide mechanisms for handling complexity in visual presentations. This complexity is primarily caused by overlap of graphical items and by visual cluttering [6]. To do so, classical interaction principles are used, as zooming, distortion and panning [7].

Further more, two other important interaction schemes are essential in the InfoVis scenario: interactive filtering [8] and Link & Brush [10]. Interactive filtering is concerned with focusing on more important regions of the data through a query facility visually carried. Link & Brush works by propagating the interactive actions of the user through various and, probably, distinct visualization scenes in order to integrate the advantages of different visual approaches.

Besides interaction, Grinstein and Ward [5] affirm that the limitations in screen resolution and color perception can be solved through multiple linked visualizations. In the present work, we mainly follow this line of research through the development of interactive mechanisms that combine diverse visualization techniques to enable scalable visual data exploration and analysis.

Hence, aiming at the limitations of visual techniques, with the aid of innovative interaction techniques, we describe our proposed systematization in the following sections of the paper. Section 2 describes the cars data set used in our demonstrations. Section 3 describes our project, the integration of ideas, the advantages achieved and a brief discussion. Section 4 formally analyses our proposal contribution. Section 5 highlights the differences between Link& Brush and our proposal. Finally, section 6 concludes the paper.

## 2 Test Data Set

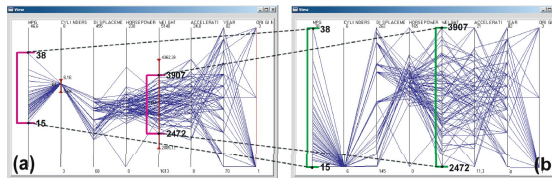
To demonstrate our proposed system, we use a version of the widely referenced cars data set (<http://stat.cmu.edu/datasets>). It has 8 dimensions and 406 data elements, that is, 3248 values to deal with. The information in the set comes from car road tests performed by the Consumer Reports magazine between 1971 and 1983. The attributes of the data are fuel performance in miles per U.S. gallon (MPG), number of cylinders in the engine (CYLINDERS), engine displacement in cubic inches (DISPLACEMENT), output of the engine in horsepower (HORSEPOWER), vehicle weight in U.S. pounds (WEIGHT), time to accelerate from 0 to 60 mph (ACCELERATION), model year (YEAR), and origin of the car (ORIGIN). The last attribute organizes the data set into three categories, American cars (1), Japanese cars (2) and European cars (3).

## 3 The Developed System

We have developed a system named *Visualization Tree*, which encompasses an environment where multiple visualization workspaces (nodes) progressively unfold into a tree-like structure. The tree can be manipulated either entirely, or individually considering each of its nodes. As the user interacts with the system, each node (visualization workspace) carries a set of data elements originated from a former node. The visual depiction of this process resembles the growth of branches from a common root. Along the text we gradually describe these functionalities, which are supported by the mechanisms described in the following sections.

### 3.1 Visualization Pipeline

The main concept applied in our system is the visualization pipeline. This interaction concept enables the data analyst to determine new visualization workspaces (nodes) based on subsets (queries) of the data. To do so, subsets of the data must be visually selected in a visualization workspace, as seen in figure 1(a). Then, based on the interactive visual query, a new visualization workspace can be defined. In this new workspace, the visual-query selected elements will be plotted in a whole new visual environment whose dimensions' boundaries are redefined according to the set of elements being investigated, as shown figure 1(b). The resulting effect presents the graphical elements expanded to a wider visual space where the details can be better noticed.



**Fig. 1.** The pipeline principle. The visual query of a previous workspace can be better observed in a whole new dedicated workspace. In the figure, the 6 cylinders cars weighting from 2472 to 3907 pounds are visually selected (a) to feed a new workspace (b).

In figure 1 we utilize the aforementioned Parallel Coordinates to illustrate the pipeline principle. From a previous scene another one can be created to stress and detail data items of interest. To perform the pipeline operation the user needs to visually select interesting items via interactive filtering. Then, to create a new workspace it is necessary to right-click the scene and choose the type of the next workspace to be fed by the data items interactively selected.

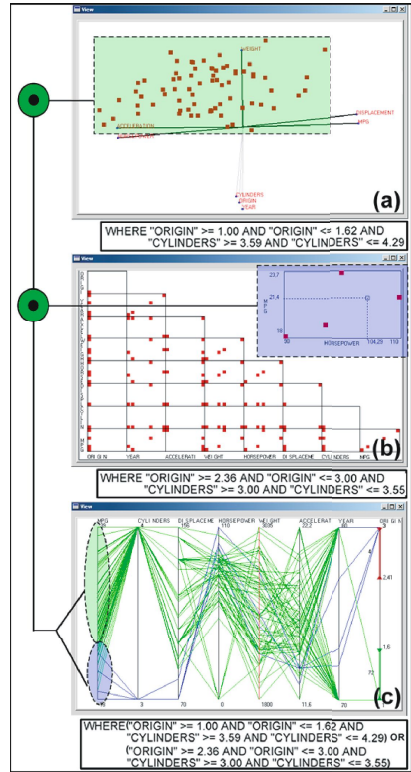
Note that this technique can be used to diminish the space limitations of a given visual technique because from overview perspective all of the scenes can be concurrently analyzed. Depending on the display size a number of scenes can be viewed simultaneously. Suppose a 17 inches (12.6x inches width) 800x600 pixels resolution display, assuming minimum observable scene size of 2x1.5 inches (usual book illustrations size). In such a display, around 6 visual scenes horizontally and around 7 scenes vertically can be arranged by means of pipelining, each of which passive of detailed inspection through zooming facilities.

### 3.2 Visualization Composition

The visualization composition stands for the combination of two or more visual workspaces, that is, it stands for the gathering of the data elements being graphically presented in each of the workspaces in a set of selected workspaces. This combination may be obtained by interactively determining which workspaces are to be considered for the composition. The composition can be performed no matter which visualization technique is being used in a given node or which technique will be used in the final composed node.

Figure 2 demonstrates the composition of two different visualization scenes. In figure 2(a) a Star Coordinates [13] node presents American cars ( $1.00 \leq ORIGIN \leq 1.62$ ) with 4 cylinders ( $3.59 \leq CYLINDERS \leq 4.29$ ). In figure 2(b), a Scatter Plots (described in [12]) node presents European cars ( $2.36 \leq ORIGIN \leq 3.00$ ) with 3 cylinders ( $3.59 \leq CYLINDERS \leq 4.29$ ). Finally, figure 2(c) presents the composition of these two figures in a Parallel Coordinates workspace. The composition is obtained by a logical *OR* and it is triggered under user request. The final visualization shows that European cars (ellipse at the bottom), even with less cylinders, are as powerful (horsepower) as American cars with 4 cylinders (ellipse at the top) at the same time that they present better performance (MPG).

The final image presented in figure 2(c) could not easily be built via conventional interactive filtering. This is an example of the ability of the composition feature, that is, the possibility of joining information initially fuzzy arranged in the data set structure. This possibility allows the creation of presentations that were previously unforeseen by the analyst who figured them out thanks to the multiple workspaces environment.



**Fig. 2.** Illustration of the visualization composition principle

### 3.3 Automatic Description Management (Labeling)

The workspaces interactively defined by pipelining and by composition carry data collections that can be straightforwardly expressed by an statement in Structured Query Language (SQL). Therefore, as an extra functionality, our system provides automatic SQL formulation in order to assist the analyst. This assistance may help to make sense of a given node and/or to keep track of the interactive process that has defined a given node. A workspace data can be described by the following SQL statement:

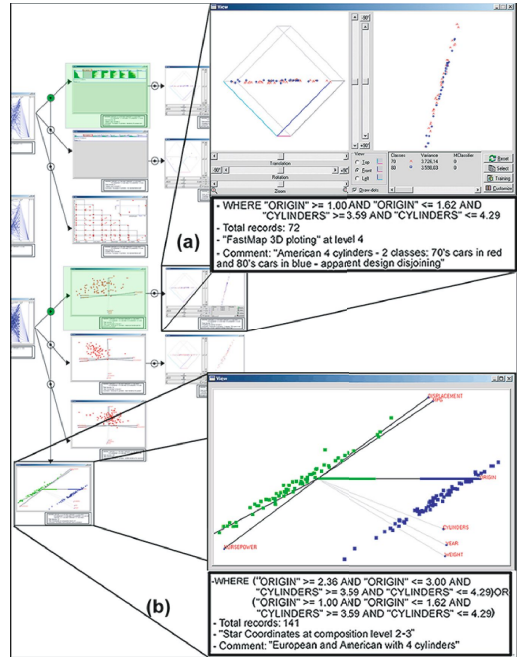
```
SELECT *
FROM ``data set``
WHERE ``interactive filtering parameters``
```

However, since all the workspaces go over the same data set and since always all the attributes are embraced by any workspace, then, the *where clause* only is enough to describe a given node. Therefore, our SQL tracking is limited to the *where clause*, which can suitably describe the data under analysis.



For each of the functionalities a different tracking is performed. For pipelining, a given workspace will carry the SQL statement that describes the interactive filtering that was defined in its father node by the time of the pipelining operation. Thus, several workspaces may be created from a common root and the automatic SQL statement will be able to describe the data set embraced by each one, as can be observed in figure 3(a). For composition, the tracking will simply concatenate the SQL statements of each of its composition nodes, interlacing them by *OR* operators (figure 3(b)).

Besides the presentation of the respective *where clause* of each node, the system also presents the total number of elements in a given workspace, the type of the visualization technique being used, the level of the workspace in the visualization tree and an optional comment that may be used by the analyst to add some conclusive observation. These features are shown in figures 3(a) and 3(b).



**Fig. 3.** The labels support the tracking of decisions performed both through pipelining (a) and through composition (b). Except for the *comment* field, all data in a given label are automatically gathered by the system providing assistance to the user during analysis.

### 3.4 Multiple Visualizations

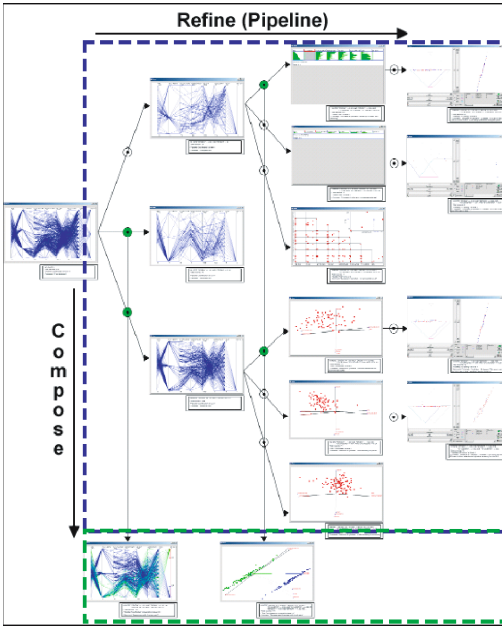
The system presented in this paper, as can be seen in figure 4, is provided with five visualization techniques, Parallel Coordinates, Scatter Plots, Star Coordinates, Table Lens [14] and a 2D/3D projection of multidimensional data sets. This projection is achieved with a dimensionality reduction algorithm named FastMap [1]. For each visualization node, the user may choose which visualization mechanism is more adequate for the presentation so that the whole environment can benefit from multiple techniques at the same time.

### 3.5 The System

In this section we describe how the concepts of pipeline, composition, automatic labeling and multiple visualizations are integrated into our system.

The principal innovation presented in this work, see figure 4, is the visual pipeline concept, which promotes progressive refinements achieved by user interaction. After

defining a selection of interest, a new visualization node can be created (pipelined) and the exploration can continue; the system mechanism is responsible for keeping track of the user-driven analysis and to present it in a tree scheme.



**Fig. 4.** An example of the visualization tree system showing three levels of visual pipeline refinement (dashed square at the top), horizontally, and two visualization compositions (dashed square at the bottom), vertically. The arrows of the tree delineate the user decisions; the labels describe the data being analyzed. The compositions embody the data sets indicated by the color-filled circular selectors.

between two levels of this tree. Pipelined scenes progressively advance toward the right part of the screen, composed scenes are positioned at the lower part of the screen. As the scene is populated, panning and zooming are necessary to focus or overview the environment.

Another important feature is the use of labels attached to the visualization nodes. These labels can be used to describe the conclusions and/or the reasoning that defined a determined branch of the tree or that defined a composition node. With the assistance of these automatic written descriptions, the exploration can be faster and thoroughly understood, even if a user had to reanalyze, later, the visual environment, or if a new user had to comprehend the findings of another former analyst.

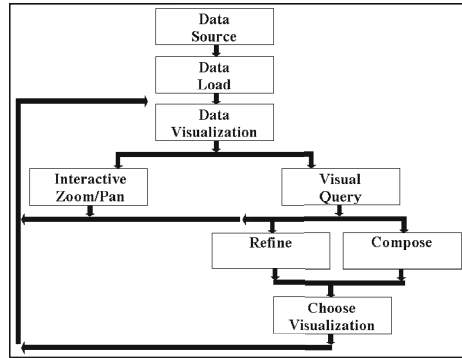
Figure 5 shows the interactive aspect of our proposed Visualization Tree system. Given a data source, each loop of the scheme determines the reading of different portions of the data. The reading is followed by the generation of a new visualization workspace or by an interactive transformation of an existing workspace. The

Along with the visualization pipeline, the system also uses the visualization composition to increase the exploratory possibilities. Compositions can be defined through the circular graphical items positioned between the levels of the tree. Each arrow of a pipelined screen carries one of these circular graphical items. The selection of one circular graphical item changes its color to green and determines that its corresponding workspace will be used for composition. This operation is illustrated in figure 4, where two compositions are defined through the mentioned circular operators. Only the color-filled circular graphical items are used for composition.

Figure 4 also presents how pipelining (refinement) and composition can be joined in a unique exploration environment. As can be seen, pipelining proceeds horizontally and composition proceeds vertically. The pipeline unfolding determines the creation of levels that constitute the browsing tree, meanwhile compositions can occur only in be-

workspaces, in turn, can be interactively explored by zoom/pan and visual querying. This last one can determine a new node in the browsing tree that will be loaded with the chosen data items.

The use of these interactive mechanisms allows the development of dynamic combinations of visualization techniques, which promotes a powerful tool for data analysis and comprehension. These techniques are enabled with visual query capabilities, present high integration and are interaction sensible. In figure 6 we demonstrate these features through 4 refinement operations. From figure 6(a) to figure 6(d) visual filtering commands are followed by pipeline operations to create new workspaces of exploration. In figure 6(d), both European (ORIGIN = 3) and American (ORIGIN = 1) cars can be visualized in Parallel Coordinates. In figures 6(f) and 6(g) the European cars are further detailed. At the final scene 6(h), European cars can be analyzed with respect to the number of cylinders in Star Coordinates workspaces at the same time that European cars and American cars can be viewed in separate workspaces.

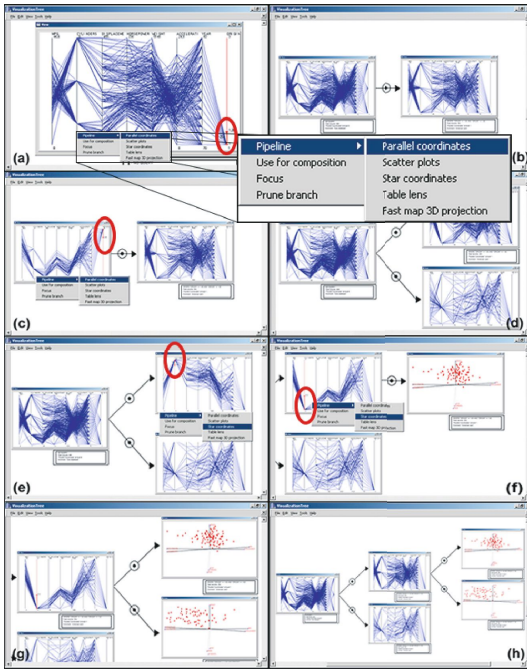


**Fig. 5.** The structure of our interaction system-atization

### 3.6 System Features

The front-end of the system is a graphical environment where visualization nodes can be embedded and manipulated by user interaction. In such graphical environment, the user is enabled to zoom the entire environment in order to have both overview and detailed perspectives. It is also possible to drag the visualization so that different, most interesting, parts of the environment can be positioned inside the limited boundaries of the display device. The main goal of the system is to allow a large range of explorative facilities that guide the analyst to a better usage of InfoVis. These facilities include

- Explorative memory*, unlike usual visualization environments, the Visualization Tree node structure allows the user to keep track of the decision steps that guided him/her to a determined visual configuration;
- Heterogeneous visualizations*, the system supports multiple visual techniques simultaneously, boosting the data analysis by the combination of the advantages that each technique incorporates;
- Unlimited detailing*, according to the pipeline concept, a derived new visual workspace is defined with dimensions whose boundaries are specified by the set of elements that fed this new workspace. Therefore, by the progressive derivation of a given workspace, the details of the graphical elements can be delimited until a single data element can be visualized in a dedicated visualization node;



**Fig. 6.** Demonstration of a refinement process. (a) Visual selection of the European (ORIGIN = 3) vehicles, (b) pipelined to a Parallel Coordinates workspace. (c) Visual selection of the American (ORIGIN = 1) vehicles, (d) pipelined to a Parallel Coordinates workspace. (e) Visual selection of the European cars with eight cylinders, (f) pipelined to a Star Coordinates workspace. Also in (f), visual selection of the European cars with three cylinders, (g) pipelined to a Star Coordinates workspace. Finally, (h) the resulting visualization tree from an overview perspective.

browsing the visual environment, from various perspectives and with specific focuses delegated by discovering interests.

## 4 Overlap of Graphical Items Analysis

In this section we formally demonstrate how the refinement promoted by our system deals with overlap of graphical items.

### Problem Formalization

Initially, we formalize the concepts of *screen-space*, *screen-space coordinates* and *graphical item*. Screen-space refers to the available display space that a given visualization can use for presentation. Screen-space coordinates refer to each position that

-*Undo functionality*, when the analyst redefines the visualization, he/she might eventually lose track of the former configuration, which might diminish the efficiency of the exploration in case of a misleading decision. The tree-like structure of the system stores the former steps taken by the user, who can undo her/his interaction.

-*Enhanced visualization*, the overall benefit of the proposed system, considering its arrangement and interactivity, is a generalized enhancement of the whole visualization process. The scalability offered by the system exceeds previous single node applications. At the same time, the interactivity that allows the use of multiple heterogeneous nodes also brings diverse new possibilities for the analyst to conduct the data analysis and mining aiming at to achieve knowledge discovery.

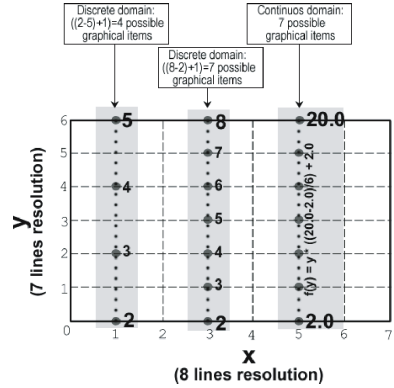
The main concept involved in the proposed Visualization Tree system is the flow of cognition through the descriptive potential of diverse visualization techniques tightly integrated in a graphical arrangement. The overall user sensation is like the possibility of

can be addressed in the screen-space. Graphical items refer to any distinct visual mark used for data visualization.

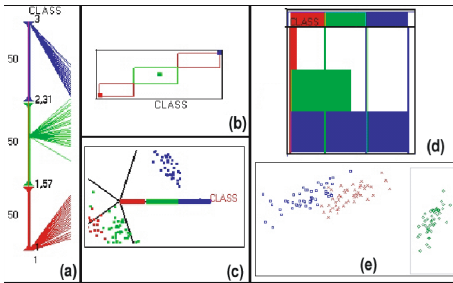
The screen-space is naturally discrete in the number of coordinates, no matter whether the domain of the data being presented is discrete or continuous. The number of coordinates in screen-space, consequently, limits the number of graphical items that can be presented. This limitation varies depending on the data domain that will be presented, as illustrated in figure 7 which uses a hypothetical one-dimensional mapping to present three different data domains. We limit our example to a one dimensional mapping, nevertheless, the same reasoning can be generalized to two and three dimensional mappings.

As demonstrated in figure 7, although the number of possible graphical items can be much bigger than the number of screen-coordinates, the number of possible graphical items cannot exceed the number of screen-coordinates. This problem is even worse for graphical items that are bigger than one pixel.

This limitation determines that visualization techniques suffer from overlap of graphical items and, consequently, by visual cluttering. Overlap of graphical items is the mapping of different data items to the same screen coordinate. Visual cluttering refers to a confused or disordered arrangement of visual items that impedes perception.



**Fig. 7.** Illustration of data with different domains mapped to screen coordinates. The figure presents three examples. The first two illustrate discrete domains and the last one presents a continuous case. Observe how the data must be adjusted to the available screen-space.



**Fig. 8.** Illustration of regions of interest. From (a) to (e) interactive definition of regions of interest via interactive filtering for Parallel Coordinates, Scatter Plots, Star Coordinates, Table Lens and 2D projection.

## Overlap of Graphical Items

For this topic, we initially formalize the concept of *region of interest*. For a given visualization technique, a region of interest is a sub-area of the screen-space in which a collection of data items is being presented. The biggest region of interest is the whole visualization itself, meanwhile, smaller regions of interest are defined via interactive filtering.

For each visualization technique, a region of interest is differently defined. As illustrated in figure 8, for Parallel Coordinates and Table Lens, it is a 1-D selection, for Scatter Plots, Star Coordinates and 2-D projection, it is a 2-D selection.

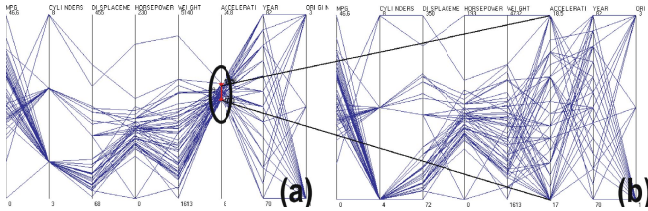
Once defined these notions, we can develop a formal metric for overlapping of graphical items. The overlap can be measured by the average number of elements that map to a same visual coordinate. Therefore, given a workspace  $\Delta$  composed of the col-

lection  $D$  of data items and composed of the set  $G$  of graphical items. The overlap factor is given by the total number of elements mapped to  $\Delta$ , that is,  $|D|$ , divided by the total number of graphical items presented in  $\Delta$ , that is,  $|G|$ . Then:

$$overlap\_factor = \frac{|data\_elements\_mapped\_to\_ \Delta|}{|graphical\_items\_presented\_in\_ \Delta|} = \frac{|D|}{|G|}$$

The same holds for regions of interest. That is, given a region of interest  $\delta \subset \Delta$ , the *overlap\_factor* is given by the total number of elements mapped to this region, that is,  $|d|, d \subset D$ , divided by the total number of graphical items presented in this region, that is,  $g, g \subset G$ .

As presented in figure 9, our pipeline principle determines that, given a region of interest  $\delta \subset \Delta$  with *overlap\_factor* =  $|d|/|g|$ , it is possible to create a new workspace  $\Delta'$  with *overlap\_factor'* =  $|D'|/|G'| = |d|/|G'|$ . Considering that a region of interest visually selected is always smaller or equal to the screen-space and considering that the workspaces created via pipelining utilize the whole screen-space, then, the following inequality is always valid  $|g| \leq |G'|$ . This is because a wider screen-space will be available for the new pipelined workspace and, thus, more graphical items can be presented. Consequently, *overlap\_factor*  $\geq$  *overlap\_factor'* is always true. In other words, the pipelining operation will always create a new scene with equal or reduced *overlap\_factor*.



**Fig. 9.** Treatment of graphical items overlap via pipeline refinement. (a) A workspace  $\Delta$  and a region of interest  $\delta \subset \Delta$  (black ellipse) defined with parameters  $17.0 \leq ACCELERATION \leq 18.5$ . (b) The workspace  $\Delta'$  achieved via pipelining the region of interest presented in (a). Also in (b), it is clear the increase in the number of graphical items, specially for the acceleration attribute. As consequence, in most dimensions not only the overlap factor was reduced but also the visual clutter in general.

## 5 Visualization Tree, Link & Brush and Other Systems

Compared to Link & Brush, our proposed views linkage does not occur at interaction level, but at decision level. This linkage is used to support the multiple visual presentations and guides the system on how to manage the several workspaces created by the user. While Link & Brush propagates interaction operations equally to different workspaces, which is what defines its views linkage, the Visualization Tree tracks the decisions took by the analyst to build a visual structure that depicts a chain of visualizations. This chain is the linkage referred by our proposed system.



It is important to stress that, different to Link & Brush, the workspaces in our system are all independent regarding interactive filtering. Their relationship is defined only by the tree structure which denotes how a given node was conceived and to which branch it belongs. The system we propose in this paper suggests a *Refine & Brush* methodology for interaction, its main advantage is the gain in number of graphical entities passive of interpretation at the same time.

Although there are other systems that implement multiple different visualization techniques in a single environment, none of them presents an interaction systematization as the one we propose. Such systems include the XGobi system [17], the Xmdv tool [16], the VisDB system [9] and the GBDIView software [15]. All of these implementations employ linked views as defined by the Link & Brush principle. For this reason, our system cannot be directly compared to these other systems. This is specially true because our main contribution is not the system itself, but the new interaction systematization which is an original contribution. Furthermore, our systematization can be adapted to any Information Visualization application and, with the aid of an appropriate control interface, it can even coexist with Link & Brush interaction.

## 6 Conclusions

We described the Visualization Tree, an interaction system that supports decision activities within the process of data analysis and exploration. Its systematization is composed of a set of new visual interaction mechanisms that exist in the context promoted by the joining of its features. These interaction mechanisms are the visualization pipeline, the visualization composition, the automatic workspace labeling, the multiple visualization combination and the environmental dragging/zooming of a user-built tree-like structure. The integration of these features resulted in an interactive extended exploratory space in which scalability of Infovis techniques is augmented.

The Visualization Tree is not to be compared to other interaction mechanisms, as details-on-demand and distortion. In fact, it can be combined to them in an environment to empower visualization techniques and interaction mechanisms of several sorts. Our approach can be used as a model to redefine the systematization of visual exploration in a scheme to diminish spatial limitations. To reach this feature we dealt with the problems of graphical items overlapping and, consequently, of visual cluttering. This aid constitutes the main contribution of this proposal.

In the same line of development that we have applied in this work, we argue that the future of Information Visualization relies on the development of enhancement mechanisms that can empower the latent capabilities of visualizations techniques. These enhancement mechanisms can be based on innovative interaction, on summarization procedures (statistical), on compounding systematizations and on improved combinations of diverse, less complex, systems. Here, we demonstrated that a main goal to be followed is to diminish the amount of information that a user must inspect at a time. In our system, we partitioned the information and allowed the user to quickly recover any portion of it, for comparison, for analysis and for putting together individual conclusive indicium.

**Acknowledgements.** This research has been supported, in part, by the São Paulo State Research Foundation Agency (FAPESP) under grant Grant #03/01144-4 and by the Brazilian National Research Council (CNPq) under grants 52.1685/98-6, 860.068/00-7 and 35.0852/94-4.

## References

1. Christos Faloutsos and K. Lin. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *ACM Int'l Conference on Data Management (SIGMOD)*, pages 163–174, Zurich, Switzerland, 1995. Morgan Kaufmann.
2. U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. Knowledge discovery and data mining: Towards a unifying framework. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 82–88, Portland, Oregon, USA, 1996. AAAI Press.
3. Jean-Daniel Fekete and C Plaisant. Interactive information visualization of a million items. In *INFOVIS*, pages 117–, 2002.
4. G. G. Grinstein, M. Trutschl, and U. Cvek. High-dimensional visualizations. In A. Keim and St. Eick, editors, *Knowledge Discovery and Data Mining - Workshop on Visual Data Mining*, San Francisco, California, USA, 2001.
5. G. G. Grinstein and M. O. Ward. Introduction to data visualization. In U. Fayyad, G. G. Grinstein, and A. Wierse, editors, *Information Visualization in Data Mining and Knowledge Discovery*, pages 21–45. Morgan Kaufmann Publishers, 2002.
6. A. Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multidimensional geometry. In *IEEE Visualization*, volume 1, pages 361–370. IEEE Computer Press, 1990.
7. Eser Kandogan. Visualizing multi-dimensional clusters, trends, and outliers using star coordinates. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 107 – 116, San Francisco, California, 2001. ACM Press.
8. Daniel A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, 2002.
9. Daniel A. Keim and Hans-Peter Kriegel. Visdb: Database exploration using multidimensional visualization. *IEEE Computer Graphics and Applications*, 14(5):16–19, 1994.
10. Daniel A. Keim and Hans-Peter Kriegel. Visualization techniques for mining large databases: A comparison. *IEEE Transactions in Knowledge and Data Engineering*, 8(6):923–938, 1996.
11. Allen R. Martin and M. O. Ward. High dimensional brushing for interactive exploration of multivariate data. In *6th IEEE Visualization Conference*, pages 271–278, Atlanta, Georgia, USA, 1995.
12. M. C. F. Oliveira and H. Levkowitz. From visual data exploration to visual data mining: A survey. In *IEEE Transactions on Visualization and Computer Graphics*, 2002.
13. Gregory Piatetsky-Shapiro and W. J. Frawley. *Knowledge Discovery in Databases*. MIT Press, Cambridge, MA, 1991.
14. R. Rao and S.K. Card. The table lens: Merging graphical and symbolic representation in an interactive focus+context visualization for tabular information. In *Proc. Human Factors in Computing Systems*, pages 318–322, 1994.
15. J.F. Rodrigues Jr., A.J. Traina, and C. Traina Jr. Enhancing data visualization techniques. In *Third IEEE Intl. Workshop on Visual Data Mining - VDM@ICDM03*, pages 97–112, Melbourne, FL, USA, 2003.
16. A. Rundensteiner, M. O. Ward, Jing Yang, and Punit R. Doshi. Xmdv tool: Visual interactive data exploration and trend discovery of high dimensional data sets. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 631 – 631, Madison, Wisconsin, USA, 2002. ACM Press.
17. D. F. Swayne, D. Cook, and A. Buja. Xgobi: Interactive dynamic data visualization in the x window system. *Journal of Computational and Graphical Statistics*, 7(1), 1998.



# Structure Determines Assignment Strategies in Diagrammatic Production Scheduling

Rossano Barone and Peter C.-H. Cheng

Representation and Cognition Group,  
Department of Informatics, University of Sussex, Brighton, BN1 9RN, UK  
r.barone@sussex.ac.uk, p.c.h.cheng@sussex.ac.uk

**Abstract.** The article discusses an application of the Representational Epistemology (REEP) approach to the development of a graphical interface for production planning and scheduling called ROLLOUT. The paper reports two studies conducted with bakery planning and scheduling professionals that focus on the cognitive support provided by ROLLOUT in reasoning about the spatial-temporal structure of production processes. The results of the first task suggest that given tabular representations, participants frequently make errors in decisions about the assignment of production sequences in terms of their temporal structure. The errors are more frequent for problems that require reasoning about consequences backwards rather than forwards in time. The second task evaluated participants' performance at improving production schedules on ROLLOUT. Log files revealed that participants improved the temporal efficiency of the schedule by making edit operations involving frequent repetitions with the same instances that tended to change the time and not the order of production. The authors propose that participants were able to effectively use ROLLOUT to generate an efficient schedule at increasing levels of precision by using the representation to test out and re-evaluate concurrent what-if scenarios. The advantages of ROLLOUT for such tasks are explained in terms of (1) the presence of diagrammatic constraints that preserve domain relations, (2) the presence of a global interpretive scheme, and (3) the degree of meaningful conceptual interrelations available in the representation.

## 1 Introduction

Representational Epistemology (REEP) is an approach to the design and analysis of external representations to support high-level cognitive activities. The REEP approach began in the design of representations to support learning and problem solving in mathematical and scientific domains [3, 5]. In recent years the approach has been applied to the development of representational systems to support real world scheduling problems [1,2,4]. One of the central concerns of the approach relates to the extent and way that an external representation encodes the underlying relational structure of the represented domain. The REEP approach to the design of external representations advocates 'structural integration' of the core systems of domain relations through analogous systems of graphical relations. We use the term structural integration to re-

fer to diagrammatic integration in which different representing objects are systematically interconnected through semantically interpretable relations that exists between their multiple properties. In addition to other traits, we have argued that the structural integration should exploit the capacity of the following diagrammatic traits (1) dependency constraints, (2) conceptual interrelations and (3) global interpretive schemes.

**Dependency Constraints.** We have used the term Law Encoding Diagrams to refer to particular classes of diagrams that have been designed to capture the underlying laws or interdependencies that exist in the domain being modelled through diagrammatic constraints of the external representation [3, 5]. Specific types of dependency constraints in diagrams have been identified. For example, free-rides refer to the capacity of a diagram to make consequential information available as a result of specifying some relation [6]. Auto-consistency refers to the capacity of a diagram to maintain that the information it expresses is logically or arithmetically consistent [8]. We consider both types of constraints important in representational design.

**Global Interpretive Schemes.** Problem solving in many domains involves the interpretation of higher-order conceptual invariants. In the case of scheduling domains the concepts of space and time are common examples. One of the goals of the REEP approach is to design representations that are structured around higher-order conceptual dimensions. Global interpretive schemes are observed in representations where there are overarching graphical dimensions for interpreting different types of entities. Such schemes involve the selection and mapping of higher-order concepts, common to different objects, to unique graphical identities [4, 5]

**Conceptual Interrelations.** These are higher-order patterns that exist between semantically specified entities and relations of the diagram. Their existence of course depends on the extent that they have perceptual characteristics that support their identification. Such patterns have been referred to as derivative meaning, as they do not or need not be specified in the representation [7]. We refer to them as conceptual because in addition to the semantic mappings of the diagrammatic entities involved, the interpretation of interrelations depends on projecting schematic knowledge involved in common forms of situational understanding (e.g. collection, numerosity, magnitude, linearity etc.). The existence of conceptual interrelations implies the capacity of a diagram to integrate different perspectives and levels of abstraction through the compositional relationship between the represented entities and patterns of interrelations. The structural integration of alternative perspectives and levels of abstraction is an important design heuristic of the REEP approach.

This article will discuss a prototype graphical interface to support bakery planning and scheduling called ROLLOUT. The ROLLOUT interface was developed under the REEP approach with an emphasis on exploiting the degree of structural integration of domain knowledge and the semantic traits discussed. We will report data on recent empirical studies with the ROLLOUT system that focuses on the cognitive support provided by ROLLOUT in reasoning about the temporal-spatial structure of

production processes. In the next section we will introduce the reader to some key issues in bakery planning and scheduling.

## 2 Bakery Scheduling

The knowledge acquired on the domain of bakery planning and scheduling is based on communication with employees of a number of bakery organisation and academic experts in the field of bakery production. The knowledge acquisition strategies have included structured interviews, responses to questionnaires, ethnographic analysis conducted in bakery organisations and analysis of external representations.

A bakery schedule is a specification of production that satisfies a variety of hard and soft constraints. The task of constructing a schedule involves determining the batches that have to be made based on orders for products. Each batch is assigned to one or more runs. Each run is a quantity of units to be produced in a sequence of processing stages using various pieces of machinery. A bakery schedule is therefore a specification of runs using particular pieces of equipment at particular times during the production period. Variation in product attributes such as ingredients, weight, size and shape, result in variation in the temporal and spatial processing values associated with the production of the particular types of products. Many of the constraints that need satisfying result from the heterogeneity of temporal and spatial processing values. The constraints listed in Table 1 represent a subset of constraints we have documented that are typical of bakery scheduling problems.

**Table 1.** Bakery scheduling constraints

Constraint name	Description
Temporal efficiency	All production should be completed in the shortest period of time
Maximal shelf life	Orders must not be produced too early for their deadline
Late completion	The deadline of an order should not be exceeded
Machine capacity excess	The capacity of production machinery cannot be exceeded
Run clash	Different runs cannot be assigned the same processing resources at the same time
Machine efficiency	Production machinery should be used continuously and at maximum capacity
Run continuity	The production of a run should be continuous and without breaks
Schedule continuity	The whole schedule should be a continuous plan of activity
Minimal Dough change	Runs made from similar dough should be grouped together in time
Product type order	Certain products should be produced before others

The kinds of representations used in bakery planning and scheduling are typically domain independent systems such as tables, lists and charts. The use of such representations is common to other scheduling domains that we have examined. These particular schemes of representing scheduling information do not exploit the potential for cognitive support in scheduling activities as a result of a number of negative representational traits. These traits include: **Fragmentation** – task relevant information is distributed over multiple displays. **Arbitrariness** – there is little or no domain specific relational structure present in such representations. **Under abstraction** – the information represented is typically low level. **Impoverishment** – the representations are not expressively rich. Taken together these factors detrimentally influence the cost of cognitive activity, the difficulty of effectively searching for improved permutations in contextually sensitive ways, as well as the potential risk of human error due to the inability to infer or recognize information relating to undesirable or impossible situations.

In order to cope with the inherent complexity of the scheduling problem a number of strategies are sometimes adopted to reduce the cognitive requirements of the problem solving tasks. However these strategies have negative trade-offs in terms of the quality and efficiency of production. Examples of such strategies include: (1) re-using previous schedules rather than exploring the potential advantages of alternative schedules; (2) employing crude heuristic guidelines (e.g. there should be  $N$  minutes between the production of each mix irrespective of the mix's processing parameters); (3) scheduling production in a short time window rather than scheduling patterns of production to accord with predicted requirements; (4) forcing all products to have the same processing times by varying factors such as the ingredients, despite the detrimental effect on product quality. This effectively eliminates the problem as the need to contend with the statistical heterogeneity of production runs has effectively been removed.

### 3 ROLLOUT System Overview

The ROLLOUT system was designed essentially as a problem solving representation to support understanding and reasoning in the process of generating plans and schedules. Other scheduling functions, such as data input and the output of low-level information, are considered peripheral issues, so have not been addressed. The ROLLOUT system consists of two main views of information. The planning view displays sets of required orders and batches. The scheduling view shows how each batch decomposes into runs. In both views, the location of all graphical objects along the horizontal axes indicates their place indexed on a timeline (see Figure 1).

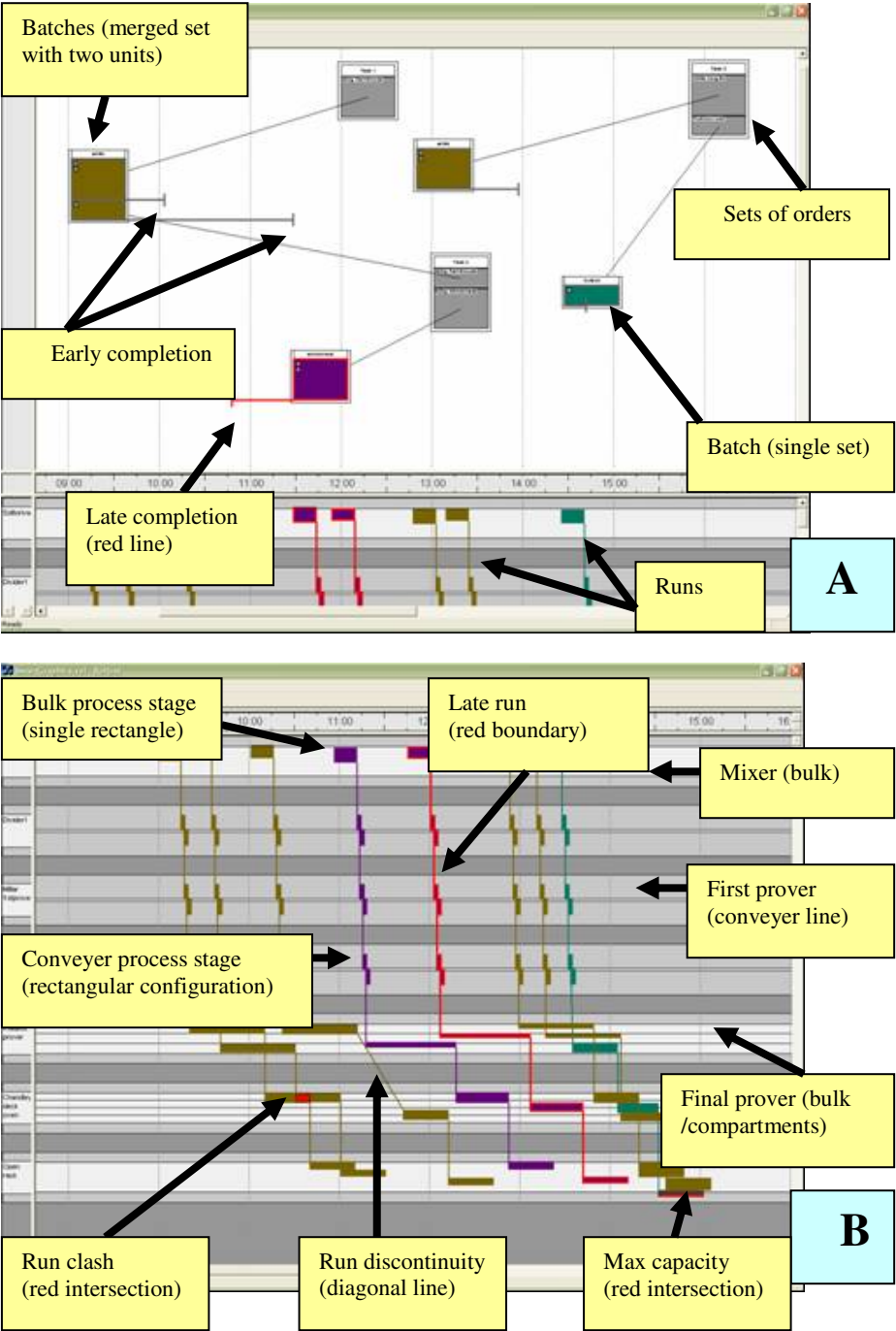
**Planner View.** The planning view displays the sets of orders that need to be produced and the batches that represent production instantiations of the order requirements. The sets of orders are shown by collections of one or more light grey rectangles enclosed in white bounding rectangles. Each grey rectangle is a particular requirement for a quantity of product. In a similar fashion sets of batches are represented by collections of one or more coloured rectangles enclosed within white

bounding rectangles. Each coloured rectangle is a batch that represents a production instantiation of single order requirement. This relation is shown by a line that connects the batch to its corresponding order. The colour of the batch rectangle represents the type of dough involved. Hence batch rectangles with similar colours can be aggregated and mixed together. Each individual batch also has a start-line whose end point indicates the ideal start time for its assignment given the required completion time of its corresponding order. A batch that is positioned to the right of its start-line indicates that it will be produced too late because the production will be completed after the order's completion time. Under such circumstances the start-line will be coloured red highlighting a serious problem. However a batch that is positioned to the left of its start-line will be completed before its required completion time. In both cases the length of the start-line indicates the magnitude of the difference between its current and required completion time.

**Schedule View.** The schedule view displays in detail how each of the batches, as shown in the planner view, are instantiated as production runs. The stages of production are represented as horizontal bars ordered from top to bottom along the vertical axes of the diagram. Each bar also contains one or more light grey bars that represent particular pieces of equipment. When creating schedules it is necessary to distinguish between two main types of equipment: (1) conveyer equipment and (2) bulk equipment. Conveyer equipment are represented by thin bars that act as timelines that quantities of product can be placed on. Bulk equipment are represented as thick bars that act of containers that quantities of product can be placed in. Some bulk equipment have sub-compartments such as shelves or cabinets. These sub-compartments are represented as subdivisions of the equipment bar. The height of the bulk stages represents the relative capacity of the particular equipment used. The full height corresponds to 100% capacity used.

Runs are represented as sequences of interconnected rectangles running diagonally from top to the bottom of the schedule. The assignment of a run to a particular piece of equipment is represented by the presence of a run rectangle in an equipment bar. There is a rectangular structure for each stage of the run but there are differences in the way process stages for bulk and conveyer type equipment are represented. Bulk process stages are shown by individual rectangles. The width of the rectangle indicates the total processing duration whereas the height of the rectangle relative to the equipment bar represents the percentage of equipment capacity that it fills. Conveyer process stages are represented as paired rectangles. The width of the two rectangles represents the time it takes the full mix to be loaded and unload on the conveyer respectively. Therefore, the total distance between the left side of the first and the right side of the second conveyer rectangles expresses the duration to process the whole mix.

These differences in the representation of run stages for bulk and conveyer processes differentially constrains the start time of the stage of a subsequent run given the position of the preceding run. In the case of bulk stages a subsequent mix can only be processed after the current mix is processed (unless there is capacity for the mixes



**Fig. 1.** The ROLLOUT Interface showing mainly the planning view (A) and only the schedule (B)

to be processed simultaneously). In contrast, for conveyer stages a concurrent mix can be assigned as soon as the preceding mix is loaded on the process. These different graphical configurations correctly differentiate between when a run clash will or will not occur for bulk and conveyer type equipment through the particular way they reveal spatial intersections. In bulk process equipment intersection occurs when the left side of a stage from a subsequent run overlaps the right side of the preceding stage. In conveyer process equipment an intersection occurs when the left side of the load-on rectangle for a subsequent stage overlaps the right side of the load-on rectangle of the stage of the preceding run (see figure 1).

ROLLOUT is an expressively rich graphical system that expresses a complex array of concepts through individual relations and patterns of interrelations. Many of these relational concepts are presented in table 2. Note that that each item can be viewed at different levels of granularity, from individual instances to groupings under self imposed or diagrammatic frames of reference.

**Table 2.** Main classes of relational concepts used for bakery planning and scheduling encoded by ROLLOUT

Meaning	Graphical expression
Minimal duration	Magnitude of spaces between successive runs
Maximal shelf life	Magnitude of distance of a left side start-line
Late completion	Presence of a right side start-line
Machine capacity excess	Run rectangles overlap in equipment region
Maximum machine capacity	Extent that the machine region is filled by the run rectangles
Run clash	Spatial intersection between stage rectangles of consecutive runs
Run continuity	Vertical alignment of successive stages
Schedule continuity	Homogeneity of spaces between successive runs in the schedule
Minimal Dough change	Extent to which proximal runs of the same colour are grouped together
Type order	Spatial order of runs in terms colour
Run temporal profile	Pattern of change in the spatial extension of runs along the horizontal axes viewed through global properties of the diagonal trajectory (e.g. steepness, curvature)
Run spatial profile	Pattern of changes in the relative spatial extension of successive stages of a run along the vertical axes
Run time duration	Overall steepness of the run trajectory
Equipment filling profile	The pattern of change in fullness along the vertical axes of a particular piece of equipment (excludes conveyer type equipment)

**Interactive Functionality.** The software has been designed so that users can selectively zoom in and out as well as show, hide or resize any of the views. Drag and drop functionality applies to sets of orders, sets of batches, individual stages, groups of stages in a run, individual runs and groups of runs. Users group runs together by drawing a lasso on the timeline with the mouse. Each run that starts in the lasso area becomes part of the group. The lasso can then be dragged left or right allowing the user to move whole sections of production forwards or backwards in time. To express that one or more batches will be produced together, batch-units can be selectively aggregated into different sets by dragging and dropping target batch-units over the desired batch-set.

## 4 User and Task Evaluation

The following tasks were performed at the Campden and Chorelywood Food Research Association (CCFRA) as part of a number of evaluation tasks performed over the course of a day. All participants ( $n=10$ ) were bakery planning and scheduling professionals (i.e. management/supervisory levels) that were either employees of instore bakeries ( $n=5$ ), plant bakeries ( $n=4$ ), or from an organisation involved in the development of bakery production technology ( $n=1$ ). All but one participant had many years of experience working in the bakery industry ( $M=24.9$ ,  $SD=12.6$ ) and were either generating production schedules in their current employment or had previously done so in the past three years. Participants were employed by organisations that were members of the research project consortium.

### 4.1 Task 1. Reasoning About Production Processes

Our assessment of the information and procedures required to construct schedules using conventional representations suggests that correctly representing and updating accurate models of the temporal-spatial structure of a schedule was a complex and difficult task. We developed a procedure to investigate the level of competence that bakery schedulers have in reasoning about particular models of production processes. The design and preliminary results of the task with scheduling professionals are discussed below.

**Design/Procedure.** Each participant was given a set of problems printed on paper. Each problem had a tabular schedule comprising of a sequence of runs of a standard product, called 'widget bread', and a single run of new product. In addition, there were also two processing templates in tabular form that specified the processing times for each stage of the new product and widget bread. The spacing between consecutive runs (inter-run duration) of widget bread was always thirty minutes, which was the minimal inter-run duration. This could be worked out from the processing template of widget bread, given it had two bottleneck stages of thirty minutes. The new product was always given a default inter-run duration of thirty minutes. The task was a within subjects design in which problems were systematically varied in accordance with three experimental factors: (1) whether the new product run was assigned before or



after the widget bread (Schedule order: Before Vs After); (2) which of the two bottleneck stages of the new product were changed in the product template (Stage selection: First Vs Last Vs Both) and (3) what kind of change was made to the duration of the bottleneck stage/s (Stage change: Shortened Vs Increased). Participants were informed of the problem situation and asked to judge whether the start time of the product should be moved earlier, later or kept the same and provide an estimation of the required change in duration.

**Results/Discussion.** Two participants were eliminated for not correctly following the procedure. Correct responses for the remaining eight participants were tabulated for each problem (duration estimations have not been included in this analysis). The fraction of individuals who made correct response is highly variable for the different conditions ( $M = .58$ ,  $SD = .26$ ). The data was entered into a  $2 \times 3 \times 2$  repeated measures analysis of variance (ANOVA). The ANOVA revealed a statistically significant main effect of correctness for schedule order [ $F(1, 7) = 31.5$ ,  $p = .001$ ] indicating that participants were more likely to make correct responses for problems in which they had to reason about the consequences of assigning a run before widget bread ( $M = .71$ ) rather than after widget bread ( $M = .46$ ). However, the main effects between the levels of stage change [ $F(1, 7) = .10$ ,  $P = .75$ ] and stage selection [ $F(2, 6) = 2.4$ ,  $P = .13$ ] were not significant. There was also a significant two-way interaction between schedule order and stage selection [ $F(2, 6) = 11.95$ ,  $P = .001$ ] revealing that more participants made correct responses about assigning the new product before the widget bread when the stage selection involved the first or both bottleneck stages. However, the reverse trend was true when the stage selection was the last bottleneck stage, hence in these situations there were more correct answers for the problems that involved assigning the new product after widget bread.

These results suggest that such tasks are inherently difficult even for individuals whose profession involves thinking about the temporal sequences. The patterns of results for the manipulation of scheduling order also suggest that reasoning about the impact of process sequences backwards in time is more difficult than sequences forwards in time. Inferring the consequences forwards in time may be more natural because of the temporal asymmetry of causal events. The interaction effect may depend on asymmetries between first and last bottleneck stages in terms of how they impact on stages of the neighbouring run as a function of their order relation.

## 4.2 Task 2. Production Scheduling with ROLLOUT

**Design/Procedure.** Participants were trained on ROLLOUT in an interactive lab session that lasted approximately forty minutes. Participants were then instructed to complete a fifteen minute practice task that involved removing constraint violations from a schedule. The practice task was followed by two scheduling tasks that required participants to improve a poor schedule. Participants were given a weighting scheme for each constraint violation in order to standardise the values of importance of constraints between people from different bakeries. All scheduling tasks were performed in groups of two, comprised of people who worked for the same organisation. Sched-

uling problems consisted of a specification of tasks and their deadline requirements together with a poor schedule solution. All scheduling problems were constructed from a data set that describes the training bakery at the CCFRA. Each problem comprised of three sets of orders requiring ten production runs.

**Results/Discussion.** This section reports data from only the first of the two scheduling problem. End solutions and interface operations were automatically recorded for each group. The percentage of improvement made to the temporal efficiency of schedule was a substantial amount ( $M=31.9$ ,  $SD=5.1$ ), which reveals that participants were actively targeting the temporal efficiency of schedule. Users made a large number of edit operations during the session ( $M=88.8$ ,  $SD=10.3$ ). Participants were more likely to manipulate production by moving runs in the schedule ( $M=69.8$ ,  $SD=11.6$ ) than batches in the planner ( $M=16.8$ ,  $SD=11.4$ ), whereas other kinds of edit operations involving the regrouping of batches were infrequent ( $M=2.2$ ,  $SD=2.5$ ). The pattern of results is not surprising as the schedule provides much more specific information on scheduling constraints. However, the significant presence of edit operations on batches in the planning view suggests that some of the decisions concerning the assignment of production were based on information in the planning view that is absent in the schedule view (e.g. ideal completion times, batch order groupings and mapping).

In order to get an measure of the kinds of changes participants were making to the schedule we categorised behaviour in terms of whether an operation changed: (1) the order of production; (2) the duration but not the order of production; (3) neither (1) or (2), which would include positioning objects vertically. The greatest percentage of operations changed the duration of production but not the order ( $M=61$ ,  $SD=11.5$ ), substantially less operations actually changed the order of production ( $M=18.5$ ,  $SD=8.6$ ), and even fewer involved neither of these ( $M=14.4$ ,  $SD=10.6$ ).

Our inspection of the log files revealed that users make sequences of operations on the same objects. We hypothesised that such sequences were associated with acts of using the representation to test the assignment of runs at greater levels of precision. If this was the case then one should be more likely to observe sequences of operations that changed the time rather than the order of runs. In order to get an assessment of this we selected operations that moved runs in the schedule or batches in the planning view, which on average accounted for approximately ninety eight percent of all edit operations. We then computed the percentage of occasions that a subsequent operation involved the same category of operation on the same token objects (e.g. run or batch). The analysis revealed that the percentage of edit operations that changed the duration but not the order of production that were categorised as a concurrent repetitions on the same object was substantial for cases when a run was moved in the schedule ( $M=30.1$ ,  $SD=19.3$ ) or when batches were moved in the planner view ( $M=31.6$ ,  $SD=13.8$ ). In contrast, the relative percentage of repetition operations that changed the order of production was minimal when runs were moved in the schedule view ( $M=4.6$ ,  $SD=4.3$ ) and absent for batches moved in the planner view.

These results suggest that participants use ROLLOUT to test out concurrent ‘what-if’ scenarios in specifying the temporal assignment of runs at increasingly precise levels of temporal efficiency.

## 5 General Discussion

The ROLLOUT interface was developed under the REEP approach to support interpretation and reasoning about bakery scheduling. The REEP approach proposes that designers should focus on trying to capture the knowledge structure of the representing domain in the external representation. Capturing the knowledge structure implies the coherent or systematic mapping of domain relations to graphical relations. The nature of the mapping should: (1) structure the representations according to the global conceptual dimensions of the domain; (2) structurally integrate alternative perspectives and levels of abstraction through the composition relations between graphical entities, relations and high-order interrelations; (3) capture the underlying dependencies of the domain through constraints in the representation.

Interviews and questionnaires conducted with bakery planners and schedulers suggest that the temporal efficiency of production is an important high-order scheduling constraint. We have provided data from task 1 conducted with bakery planning and scheduling professionals revealing that reasoning about high-order relations between temporal sequences is difficult and that this difficulty may be more pronounced for inferences made on events backwards rather than forwards in time. The analysis of log files and solutions in task 2 revealed that participants were able to generate a time efficient schedule at increasing levels of precision by using the representation to test out and re-evaluate concurrent what-if scenarios.

Confronted with tabular style representation of production and processing parameters of products the task of finding a precisely derived efficient schedule is arithmetically complex. It is not surprising that some schedulers may resort to estimations based on coarse heuristics. Exactly what kinds of properties of the ROLLOUT representation support these kinds of precision directed strategies in scheduling? We argue that these strategies depend on the three semantic traits discussed in the first section: (1) constraint dependencies, (2) conceptual interrelations and (3) global interpretive schemes.

**Constraint Dependencies.** The user needs to correctly infer how relations between different objects change as a result of a change in another. One important example of constraint dependencies is present in the relational structure of a run. As the positions of stages  $N+1$  or  $N-1$  are always dependent on the position of stage  $N$ , the relational system captures how the whole run will change as consequence of a change in the temporal assignment of a stage. By being able to comprehend these structural constraints the users is able to behave in more informed ways in acts of look-ahead.

**Conceptual Interrelations.** In order to specify the assignment of neighbouring runs the user needs to consider not one or several relations but patterns of interrelations that exist between neighbouring runs. Consider the conceptual interrelations between

consecutive runs, such as the patterns of durations between the stages of concurrent runs, the ordinal pattern of the duration between sets of stages of consecutive runs, the relative percentage capacity of spatial filling on a machine, etc. The space of conceptual interrelations available in ROLLOUT allows the user to select interrelations that are relevant to the particular task and contextual characteristics of the situation under consideration. This allows the user to assign production in a more highly informed way.

**Global Interpretive Schemes.** The availability or capacity to identify conceptual interrelations depends on perceptual as well conceptual/semantic factors. Conceptual interrelations are easier to comprehend when there is a coherent overarching scheme supporting their interpretation. In ROLLOUT there is a global interpretive scheme for reasoning about spatial and temporal values of schedule entities and relations (Time → Horizontal, Space → Vertical). We argue that the selection of relevant interrelations in specifying assignments is supported by the global interpretive scheme because recognising and interpreting task relevant interrelations depends on the ease with which the cognitive system correctly maps graphical patterns to conceptual interpretations. The rules that the cognitive systems follows for such mappings will be simplified if the external representation has a global interpretive scheme.

Scheduling representations should be rich in terms of dependency constraints and conceptual interrelations as well as being structured according to the global conceptual dimensions of the represented domain. These traits influence numerous tasks. An example has been provided involving reasoning about assignments based on the spatial-temporal structure of production processes. Ongoing research with ROLLOUT will be aimed at extending and validating these claims.

## Acknowledgements

The research was supported by an ESRC research grant (RES-328-25-001) under the PACCIT programme. We would like to acknowledge Nikoleta Pappa for valuable comments on earlier designs of ROLLOUT, Stan Cauvan and Linda Young for sharing with us their expertise of the baking industry, and all three for their efforts in the realization of the experiment reported here.

## References

- [1] Barone, R., Cheng, P. C.-H., Ahmadi, S., & Cowling, P. I., (2003). The strategic influence of conceptual structure in graphical interfaces for scheduling. In P. G. T. Healey (Eds.), *Interactive Graphical Communication Workshop 2003. Working Papers*. Queen Mary: University of London, 7-20.
- [2] Barone, R., Cheng, P. C.-H. (2004). Representations for Problem Solving: On the Benefits of Integrated Structure. In E. Banissi, K. Borner, C. Chen, M. Dastbaz, G. Clapworthy, A. Faiola, E. Izquierdo, C. Maple, J. Roberts, C. Moore, A. Ursyn, J. J. Zhang (Eds.), *Eighth International Conference on Information Visualization, IV04* (pp. 575-580). IEEE Computer Society.

- [3] Cheng, P. C-H. (1999) Unlocking conceptual learning in mathematics and science with effective representational systems. *Computers in Education*, 33(2-3), 109-130.
- [4] Cheng, P. C.-H., Barone, R., Cowling, P. I., & Ahmadi, S. (2002). Opening the information bottleneck in complex scheduling problems with a novel representation: STARK diagrams. In M. Hegarty, B. Meyer, & N. H. Narayanan (Eds.), *Diagrammatic representations and inference: Second International Conference, Diagrams 2002*. Berlin: Springer, 264-278.
- [5] Cheng, P. C-H. (2002). Electrifying diagrams for learning: principles of complex representational systems. *Cognitive Science*, 26, 685-736.
- [6] Shimojima, A. (1996). Operational Constraints in Diagrammatic Reasoning. In: G. Allwein and J. Barwise (Eds.): *Logical Reasoning with Diagrams*. Oxford: Oxford University Press, pp. 27-48.
- [7] Shimojima, A. (1999). Derivative meaning in graphical representations. *Proceedings of 1999 IEEE Symposium on Visual Languages*, 212-219.
- [8] Stenning, K., Inder, R. & Neilson, I. (1995). Applying semantic concepts to analyzing media and modalities. In: Glasgow, J., Narayanan, N. H., & B. Chandrasekaran, B., (Eds.): *Diagrammatic Reasoning: Cognitive and Computational Perspectives*. AAAI Press Menlo Park, CA, 303-338.

# Generation of Glyphs for Conveying Complex Information, with Application to Protein Representations

Greg D. Pintilie<sup>1</sup>, Brigitte Tuekam<sup>1</sup>, and Christopher W.V. Hogue<sup>1,2</sup>

<sup>1</sup> Blueprint, Samuel Lunenfeld Research Institute, Toronto, Canada  
{gpintilie, btuekam, chogue}@blueprint.org

<sup>2</sup> Department of Biochemistry, University of Toronto, Toronto, Canada

**Abstract.** We present a method to generate glyphs which convey complex information in graphical form. A glyph has a linear geometry which is specified using geometric operations, each represented by characters nested in a string. This format allows several glyph strings to be concatenated, resulting in more complex geometries. We explore automatic generation of a large number of glyphs using a genetic algorithm. To measure the visual distinctness between two glyph geometries, we use the iterative closest point algorithm. We apply these methods to create two different types of representations for biological proteins, transforming the rich data describing their various characteristics into graphical form. The representations are automatically built from a finite set of glyphs, which have been created manually or using the genetic algorithm.

## 1 Introduction

Humans use symbols for communication of complex information in many contexts. One example is written language, where every letter or combination of letters typically represents a sound used in spoken language. Using these building blocks, words, sentences, and papers can be generated which convey a wealth of information. Another example is the use of symbols in circuit diagrams. Here a symbol represents an electrical component with well-defined physical properties, which can be connected to other components to create a circuit diagram, from which the functioning of a complex electronic device can be deduced. Circuit diagram symbols could be deemed less general than language symbols; for example they could not be used to convey the same information that a word or a sentence does. On the other hand, a circuit diagram can be described using only words and sentences, without the loss of information, albeit at the cost of losing its conciseness. Our aim in this paper is to present methods that allow the same expressive power of graphical symbols to be used in other applications, to replace or at least complement written language as the only form of dissemination of information.

A lot of applications already use graphical symbols, a testament to their usefulness in conveying information. In our application we would like to go a bit

further and allow graphical symbols to be seamlessly combined, in the same way that words are combined to create sentences, so that more complex information can be conveyed. There are great advantages to taking this extra step. For example, individual symbols for electronic components would be rather useless if they couldn't be connected together to form complete electronic circuits. In the methods we have developed, we have focused on enabling this approach, whereby various graphical symbols, or glyphs, representing concise information on their own, can be combined to express the properties or behaviour of more complex entities.

In developing these methods, we have focused on the communication of information about large organic molecules such as proteins, which are the building blocks of the cells in biological organisms. A wealth of information is available for proteins, for example their structural form, where in (or close to) a cell they are typically found, what their function and behaviour is, what other molecules they commonly bind to, and what subcomponents they are built from. Development of applications where such information is presented has been increasing, and we imagine that it would be very useful to have methods that allow the information to be conveyed in a symbolic yet concise way through graphical representations. We hope that developing such methods for one such specific application can lead to similar methods being exploited in both biology and non-biology settings.

## 2 Overview and Related Work

The use of glyphs for conveying quantitative information is common. For example, Wittenbrink et al. use varied glyphs that illustrate the uncertainty in magnitude and direction within vector fields [25]. Rose and Wang similarly use a method that creates dense iconic plots so that the user may discern patterns in multivariate data [20]. Automatic generation of circular glyphs that communicate information about individual textures is explored by Battiatto et al. [4].

Creating glyphs to communicate more abstract information that cannot be quantified seems to be a process that requires more human involvement. Ribarsky et al. have developed an interface that allows users to create glyphs that can be linked to specific 'domain knowledge' [19]. In our approach, the glyph geometry can also be specified by the user, though the specification is done using a context free grammar similar to the one used in L-systems. L-systems were introduced by Lindenmayer to describe biological organisms [12], and later used by Prusinkiewicz in the generation of plant geometries [17]. They can also be used to describe fractals, and the variety of geometries that can be generated using such systems is immense.

A glyph is simply created and stored as a string, in which characters specify the various geometrical operations that create the glyph's geometry. It is thus easy to create new geometries simply by changing or concatenating such strings. We exploit this form within a genetic algorithm that can create a large set of diverse glyphs with minimal effort on the part of the user. Genetic algorithms have been previously used for the generation of geometrical forms by Sims [22],

as well as for other design problems [9,18]. In our work, the many resulting geometries are automatically filtered into a library of distinct glyphs that have the desired visual characteristics.

To automatically compare glyph geometries, we have looked at common geometry comparison methods. There are variety of such methods, and a good survey is provided by Alt and Guibas [1]. Most of the typical distance measures such as the Hausdorff distance, don't typically give a good measure of how visually distinct two geometries are. A better approach is to use the iterative closest point (ICP) algorithm [6]. This algorithm tries to find the best alignment between two geometries such that the sum of the squared distances between a number of corresponding sample points is minimized. We use a normalized form of this sum as a measure of the visual distinctness between two glyphs.

Having a quantitative measure that ranks the visual distinctness between glyphs also allows us to organize them in clusters according to their visual appearance. There are various clustering algorithms [7,11]; for example, one common approach for clustering points is to assign a point to the cluster whose mean it is closest to. Because it would be hard to compute the mean of a number of geometries, we cannot use this approach. Instead we use the approach of Voorhees [23], where clusters are formed based on average distances between the individuals in each cluster.

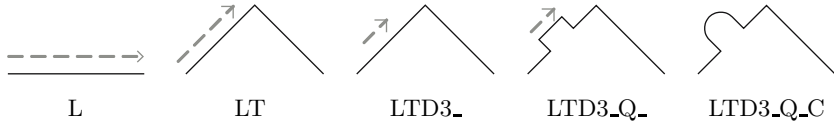
We apply these methods to create graphical representations for proteins that communicate their various properties. The most direct way to graphically portray a protein is to draw its molecular structure, as many programs do [5,14,24]. These detailed representations are useful in many ways, however they are too cumbersome to use when we just want to show the general characteristics of the protein. A simpler glyph representation would be easier to understand and comprehend all at once. Moreover the glyph representation could also be used to portray other knowledge about the object that is not obvious from its physical description.

Symbolic representations of proteins are already currently used in some applications [3,13,21]. These applications diagram the building blocks that proteins consist of, however they use very few geometrical shapes and colors. We explore ways to include more information into similar graphical representations. We have created two different types glyphs for representing proteins, which we have named *ontoglyphs* and *proteoglyphs*. Ontoglyphs make use of annotations for proteins in the form of terms from the Gene Ontology dictionary [8]. Proteoglyphs on the other hand communicate information about the structural components of the protein. Ontoglyphs and proteoglyphs are built by combining a number of smaller sub-glyphs, which are either hand-designed by users or generated by a genetic algorithm.

### 3 Glyph Geometry

In this work we focus on glyphs that are composed of a single continuous line that can take any 2-dimensional geometry, and can be either closed or open.





**Fig. 1.** A glyph is generated using geometrical operations. L creates the first line segment. Each new segment (dotted line) is affected another operation which follows: T triangle, D3 divide into 3, \_ skip, Q quadrilateral, C circle.

Internally, each segment in a glyph is represented using a Bezier curve [10], thus the glyph can contain both straight and curved segments. The glyph geometry is specified as a string, in which characters are used to represent geometrical operations. Figure 1 diagrams how a glyph is constructed in this way.

In the bracketed string notations of Lindenmayer [12], a glyph string has the following form:

$$a := Lp_l(b) \quad (1)$$

$$b := c(b^n) | d \quad (2)$$

$$c := Tp_t | Cp_c | Qp_q | Dp_d | B | Z \quad (3)$$

$$d := _ | Rp_r | Sp_s \quad (4)$$

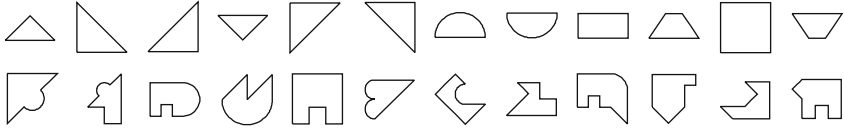
The glyph string must start with the line operation L (1), to which another operation is applied from (2). An operation taken from (3), e.g. triangle T, circle C, quadrilateral Q, divide D, back-segment B, or zig-zag Z, replaces the line segment with  $n$  new segments, to which (2) is recursively applied in succession. Alternatively, an operation from (4) may be applied to a segment, such as \_ which leaves it as it is and skips to the next segment, R which creates reflecting geometry (which cannot be further modified), or S which shortens the segment from the beginning or end. The parameters  $p_l, p_t, p_c, \dots$  specify values used by the geometrical operations, though they may be omitted in which case default values are used. All the operations and their parameters are listed and described in the Appendix. Note that the brackets in equations (1-4) are not needed since the grammar is unambiguous, and are usually left out of the actual glyph strings for simplicity.

## 4 Glyph Design Using a Genetic Algorithm

Genetic algorithms mimic the evolutionary process observed in nature, which creates a diverse population of individuals that tend to become more complex over time. A genetic algorithm involves mutation and combination of genotypes, where the genotypes contain the code necessary to create living organisms or phenotypes. Due to natural selection, only phenotypes that are fit survive. With this pruning process, nature has found a way to create many diverse individuals without exhausting every possible genotype, instead only propagating the genotypes that are likely to create fit phenotypes.

$$\text{LB}(\text{C}\text{---})^{(1)} + \text{LBT}(\text{---})^{(2)} = \text{LBTC}\text{---}$$

**Fig. 2.** A complete geometrical operation taken from one parent glyph (1) is introduced into another (2) to create a new 'offspring' glyph



**Fig. 3.** Glyphs in the starting population are in the top row. The bottom row shows some of the resulting glyphs after  $\sim 300$  steps with the genetic algorithm.

Using the same evolutionary approach, we can create many glyph geometries that are varied and satisfy some desirable fitness criteria. For this we imagine that glyph strings are genotypes and that glyph geometries are the resulting phenotypes. Starting with a set of varied, user-specified genotypes, new genotypes are created by combining random pairs of existing genotypes. The combination process is illustrated in Fig. 2.

The glyph resulting from a combination operation is added to the population if it meets some fitness criteria. We define the fitness of a glyph based on its visual appearance: a glyph is considered fit if all its segments are non-intersecting, and if non-connected segments are at least some minimum distance  $d$  from each other. These criteria are meant to ensure that the resulting glyphs have the form of a single continuous line that doesn't intersect itself. The constant  $d$  is chosen depending on the size of the glyphs being created; for example if the glyphs are to fit into a square measuring 100x100 units, choosing  $d = 5$  results in glyphs having geometries that don't appear to intersect, even when scaled down substantially. Figure 3 shows an example starting population and some of the resulting geometries.

## 5 Comparison of Glyph Geometries

The genetic algorithm can generate a very large population of glyphs, and even if every genotype is different, it is possible that some phenotypes will be similar and possibly even identical (e.g. two stacked triangles form a quadrilateral). To prune similar glyphs from the population, a measure that computes the visual distinctness of two glyphs is needed. Such a measure would be equal to zero if the glyphs are visually identical, or otherwise take a value that increases proportionally with the dissimilarity between the two glyphs. To compute this similarity measure, we use the iterative closest point (ICP) algorithm [6].

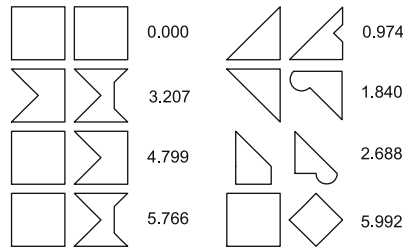
The ICP algorithm computes the alignment between two geometries which is optimal (in a least-squares sense). An initial alignment is performed such that the centers of the the two geometries coincide. A set of points is created by point-sampling one of the geometries, and the closest point on the other geometry to each sample point is found and added to a second set of points. The distance between closest points represents a local misalignment, or dissimilarity, between the two geometries. A normalized distance measure between the two glyphs can thus be computed using the root mean square deviation (RMSD) formula:

$$d(g_i, g_j) = \sigma_{ij} = \sqrt{\frac{1}{n} \sum_{k=1}^n (p_{ik} - p_{jk})^2}, \quad (5)$$

where  $g_i$  and  $g_j$  are the two glyphs being compared,  $p_{ik}$  and  $p_{jk}$  are corresponding closest points (one on each glyph), and  $n$  is the number of corresponding points.

After the initial alignment, a transformation is found which when applied to one of the glyphs decreases (on average) the distance between corresponding points. After computing the new set of corresponding closest points for the same initial sample points, the process is repeated as long as the RMSD decreases significantly with each iteration. The original ICP algorithm also considers rotational transformations that decrease the RMSD, however for our purposes this is not necessary. This is because we consider two identical glyphs with different orientations to be visually distinct, since glyphs are always displayed with the same orientation.

Figure 4 shows several glyph comparisons along with the computed RMSD. The number of sample points in each comparison can vary, since more complex geometries require more sample points, however because the RMSD value is normalized this value is factored out of the overall score. Thus pairs of glyphs can be ranked based on how similar they are, regardless of their complexity.



**Fig. 4.** Comparison of pairs of glyphs using the RMSD as a measure of visual distinctness. The RMSD is the number to the right of each pair of glyphs. In both columns, the similarity between pairs of glyphs decreases from top to bottom while the RMSD value increases.

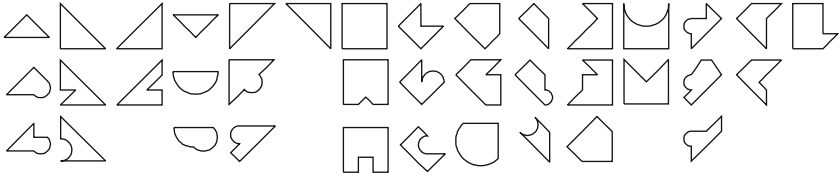
## 6 Clustering of Glyph Geometries

The distance measure between glyphs can be used to automatically organize a large number of glyphs into clusters. Doing this allows us to observe which glyphs in a large set are similar to each other. It is also useful when the number of glyphs needed is smaller than the size of the set, since by selecting only one glyph from each cluster, glyphs that are more distinct from each other can be used.

Our method creates a desired number of clusters from a large number of glyphs, so that glyphs in one cluster are on average more similar to each other than to glyphs in other clusters. The algorithm, based on the work of Voorhees [23], first creates a cluster for every glyph, and then successively merges clusters until the desired number of clusters is reached. The clusters to be merged are picked so as to minimize the average-link cost function:

$$c_{AL} = \frac{1}{|C_1||C_2|} \sum_{g_i \in C_1} \sum_{g_j \in C_2} d(g_i, g_j), \quad (6)$$

where  $d(g_i, g_j)$  is the distance measure between two glyphs  $g_i$  and  $g_j$ , defined in Eqn. (5), with each glyph being from two different clusters  $C_1$  and  $C_2$ .  $|C_1|$  and  $|C_2|$  refer to the sizes of the two clusters. As Eqn. (6) indicates, to compute the average-link cost of merging two clusters, we sum the distances between every possible pair of glyphs, one from each cluster, and divide this value by the total number pairs of glyphs considered. Figure 5 shows clusters created using this approach. Other cost functions could also be used, such as the minimum or maximum distance between any two glyphs from different clusters, which would give slightly different results [7].



**Fig. 5.** Clusters of glyphs taken from a population generated using the genetic algorithm. Each cluster is shown on a separate column. Note that it is possible that some clusters contain only one glyph, if no other glyphs in the population are sufficiently similar in geometry.

## 7 Applications

### 7.1 Ontoglyphs

An ontoglyph is used to communicate *ontological* information about the protein it represents. Information about each protein is available as a list of annotations,

where each annotation is a term from the Gene Ontology dictionary [8]. Gene Ontology terms relate to one another through parent-child relationships that form a directed acyclic graph (DAG), which can also be seen as a hierarchy where a child can have more than one parent. As the hierarchy is traversed outward from the root, the terms become more and more specific; conversely traversing toward the root we would encounter more and more general terms.

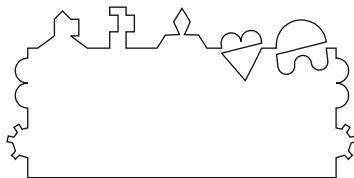
To graphically represent such annotations, we use a small number of categories, where each category is assigned one or more GO terms and a unique user-designed glyph. The glyph is designed so that it is very distinct, and so that it portrays the information associated with the category it represents in an intuitive way. Alternatively, it should have a form that users already associate with this information. To create an ontoglyph, the GO graph is searched with each annotating term for its closest ancestor which belongs to a category. (The ancestor term encompasses the same information as the annotation, albeit in a more general way.) This results in a list of categories that best represent the protein's annotations, within the expressive power of the set of categories previously defined.

The ontoglyph is built by inserting the glyphs of the computed categories into the segments of a rectangle. The categories for an ontoglyph are divided into three classes: localization, binding (or specificity), and function. The glyphs belonging to localization categories are embedded into the left segment of the rectangle, while the binding and function glyphs are embedded into the top segment. Thus the ontoglyph is created with a string that has the form:

$$\text{LBQDn}S_1..S_n\text{Dm}T_1..T_m\text{R}, \quad (7)$$








where  $S_1..S_n$  are strings for the glyphs to be embedded into the side segment and  $T_1..T_m$  are strings for the glyphs to be embedded into the top segment. Note that the glyphs embedded into the left segment are reflected onto the right segment by the 'R' operation, and the 'B' operation creates the bottom segment of the rectangle. Figure 6 shows an example ontoglyph.

For a complete list of categories and glyphs that can be included into an ontoglyph, and the biological information that each one communicates, we refer the reader to the web site: <http://seqhound.blueprint.org/cgi-bin/ontoglyphs.pl>. Table 1 lists the glyphs and categories used in the ontoglyph in Fig. 6. Because the glyphs are always the same for each category, once the user has learned



**Fig. 6.** Ontoglyph for the protein with Gene Identifier (GI):29466. The sub-glyphs along the side and top segments communicate ontological information about the protein.

**Table 1.** Some of the categories and glyphs for representing information in ontoglyphs

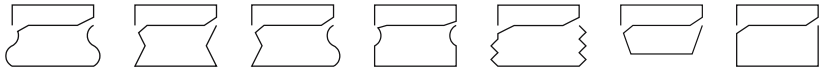
Binding	Function	Location
 Calmodulin binding	 Development	 Actin cytoskeleton
 Cytoskeletal protein binding	 Cell motility and structural activity	 Cytoplasm
 ATP binding		

this basic alphabet, the characteristics of any protein can be discerned from its corresponding ontoglyph. The total number of categories is kept relatively small (at about 84), and so while an ontoglyph may not represent the information contained in the annotations for a protein exactly, it still provides a general view which can be very useful.

7.2 Proteoglyphs

Proteoglyphs are used to show any subcomponents that a protein may contain. The largest subunits within a protein are known as *conserved domains*, and each domain has its own structural and binding properties. In a proteoglyph, each domain is shown with a separate glyph which communicates these structural and binding properties. A domain can belong to one of several structural classes based on [16]. Each class is associated with a base glyph, which illustrates the properties of the class, and which is used to construct the domain glyph. These classes are listed in Fig. 7. Binding properties for a domain are available in the form of GO terms, and as for ontoglyphs, a number of binding categories can be used to represent this information.

There are some ten thousand conserved domains in our database, and we have generated a unique geometry to represent each one. We have used the



**Fig. 7.** Base glyphs for representing conserved domains. The left and right segments in the bottom half of each base glyph are different depending on the class. From left to right, the classes are: alpha, beta, alpha+beta, alpha/beta, membrane, small, and structure unknown. The shape of the segments mimics the physical structure: alpha structures are helix-like, and thus circular segments are used; beta structures are sheet-like, and thus straight line segments are used. Classes encompassing both structures use both curving and straight segments. The zig-zags in the base glyph for membrane domains mimics how membranes are usually depicted in diagrams, and the base glyph for small domains has smaller bottom segments.

genetic algorithm to generate them and the geometry comparison and clustering techniques to ensure they are all visually distinct. The unique geometry can effectively show whether two domain glyphs shown side by side refer to different domains, even if they have the same binding and structural characteristics. Also, we think that users may be able to identify a domain they are familiar with just from its unique geometry, much as we readily recognize familiar faces amongst many unfamiliar ones.

To create a domain glyph, the binding glyphs are inserted along the top segment of the base glyph corresponding to its class, similarly to how they are inserted into an ontoglyph, and the unique geometry for the domain is added to the bottom segment. Once the domain glyphs for a protein have been determined, the complete proteoglyph can then be constructed. A proteoglyph is composed of a single line segment which mimics the form of a protein, which is simply a chain of smaller molecules known as amino acids. Glyphs representing conserved domains are inserted along this line at their respective positions, with each glyph having a width matching the length of the domain that it represents. A proteoglyph string has the following form:

$$\text{LLrD}(n * 2 + 1)\text{Ls}(\text{Ll}_1\text{Li}_1..\text{Ll}_n\text{Li}_n)\text{-(G}_1..\text{G}_n\text{)}, \quad (8)$$

where  $r$  is the number of residues the protein contains,  $n$  is the number of domains,  $s$  is the residue at which the first domain starts,  $l_1..l_n$  are the lengths of domains  $1..n$ ,  $i_1..i_n$  are the number of residues between a domain and the next domain or the end of the sequence, and  $G_1..G_n$  are the glyphs for domains  $1..n$  respectively. An example proteoglyph is shown in Figure 8.



**Fig. 8.** Proteoglyph showing 4 conserved domains in a protein (GI:1081), their positions in the protein and relative sizes. Each domain glyph shows what structural class the domain belongs to and its binding properties when known. It can be seen that all the domains are different since their unique geometries differ.

## 8 Conclusion

We have presented methods for creating graphical symbols, or glyphs, which communicate various characteristics of complex entities such as proteins. Glyphs are created and stored as strings, so combining them together into more complex glyphs is a straightforward process. We have introduced two new graphical representations for proteins, ontoglyphs and proteoglyphs, which make use of

an alphabet of sub-glyphs as well as a large set of unique glyphs to graphically communicate some of the rich information associated with such entities. Both representations are already used in the web interfaces of BIND [2], which is a database of protein interactions, and Seqhound [15], which stores other types of biological information. Ontoglyphs are also used in a graphical application supported by BIND, the Bind Interaction Viewer (BIV), which diagrams protein interaction networks.

We believe that once users have learned the basic geometrical properties of ontoglyphs and proteoglyphs, and the sub-glyphs that are used to create them, they can quickly get an idea of a protein's functional, localization, behavioral, and structural characteristics from these representations. Because sub-glyphs intuitively depict the characteristics they communicate, and also having used symbols which biologists are already familiar with, we also believe that this system would not take a long time to become familiar with. A more concrete measure of its effectiveness is yet to be determined. While this is pursued, we hope that the methods we have presented can be further applied and developed to create more effective graphical communication tools.

## Acknowledgements

This work was supported by grants to C.W.V.H. from Genome Canada through the Ontario Genomics Institute, from the Canadian Institute of Health Research and the Ontario R&D Challenge fund.

## References













1. Alt, H. Guibas, L. J.: Discrete Geometric Shapes: Matching, Interpolation, and Approximation. In: Sack, J. R., Urrutia, J. (eds.): *Handbook of Computational Geometry*. Elsevier Science Publishers, Amsterdam, Holland, B.V. North (1999) 121–153
2. Alfaro, C., Andrade, C.E., Anthony, K., Bahroos, N., Bajec, M., ...: The Biomolecular Interaction Network Database and related tools 2005 update. *Nucleic Acids Res.* **33**(Database Issue) (2005) D418–D424
3. Bateman, A., Birney, E., Cerruti, L., Durbin, R., Eddy, S.R., Griffiths-Jones, S., Howe, K.L., Marshall, M., Sonnhammer, E.L.L.: The Pfam Protein Families Database. *Nucleic Acids Research* **30**(1), (2002) 276–280
4. Battisto S., Gallo G., Nicotra, S.: Glyph Representation of Directional Texture Properties. *Journal of WSCG* **10**(1-3) (2002) 48–54
5. Bergman, L. D., Richardson, J. S., Richardson, D. C., Brooks, J.F.P: VIEW - an exploratory molecular visualization system with user-definable interaction sequences. *Proceedings of SIGGRAPH* **71** (1993) 117–126
6. Besl, P. J., MacKay, N. D.: A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14** (1992) 239–256
7. Fasulo, D.: An Analysis of Recent Work on Clustering Algorithms. Department of Computer Science & Engineering, University of Washington (1999)
8. The Gene Ontology Consortium: Gene Ontology: tool for the unification of biology. *Nature Genetics* **25** (2000) 25–29



9. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading Massachusetts (1989)
10. Hoffmann, C.M.: Geometric and solid modeling: an introduction. Morgan Kaufmann Publishers Inc., San Francisco, California (1989)
11. Jain, A.K., Dubes, R.C.: Algorithms for clustering data. Prentice-Hall (1988)
12. Lindenmayer, A.: Mathematical models for cellular interaction in development, Parts I and II. *Journal of Theoretical Biology* **18** (1968) 280–315
13. Marchler-Bauer, A., Panchenko, A.R., Shoemaker, B.A., Thiessen, P.A., Geer, L.Y., Bryant, S.H.: CDD: a database of conserved domain alignments with links to domain three-dimensional structure. *Nucleic Acids Research* **30**(1) (2002) 281–283
14. Martz, E.: 3D Molecular Visualization with Protein Explorer. In: Krawetz, S.A., Womble, D.D. (eds.): *Introduction to Bioinformatics*, Humana Press, Totowa NJ (2003) 565–586
15. Michalickova K., Bader G.D., Dumontier M., Lieu H., Betel D., Isserlin R., Hogue C.W.: Seqhound: biological sequence and structure database as a platform for bioinformatics research. *BMC Bioinformatics*, **3**(1), (2002) 32–45
16. Murzin, A.G., Brenner, S.E., Hubbard, T., Chothia, C.: SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.* **247** (1995) 536–540
17. Prusinkiewicz, P., Hanan, J.: Visualization of botanical structures and processes using parametric L-systems. In: *Scientific Visualization and Graphics Simulation*. J. Wiley & Sons, Chichester (1990) 183–201
18. Renner G., Ekrt A.: Genetic algorithms in computer aided design. *Computer-Aided Design* **35**(8) (2003) 709–726
19. Ribarsky, W., Ayers, E., Eble, J., Mukherjea, S.: Glyphmaker: Creating customized visualizations of complex data. *IEEE Computer* **27**(7) (1994) 57–64
20. Rose, S.J., Wong, P.C.: DriftWeed: a visual metaphor for interactive analysis of multivariate data. *Visual Data Exploration and Analysis VII (Proceedings of SPIE)* **3960** (2000) 114–121
21. Schultz, J., Copley, R.R., Doerks, T., Ponting, C.P., Bork, P.: SMART: A Web-based tool for the study of genetically mobile domains. *Nucleic Acids Res* **28** (2000) 231–234
22. Sims, K.: *Evolving Virtual Creatures*. Computer Graphics (1994) 15–22
23. Voorhees, E. M.: Implementing Agglomerative Hierarchic Clustering Algorithms for use in Document Retrieval. *Information Processing & Management* **22** (6) (1986) 465–476
24. Wang Y., Geer L.Y., Chappay C., Kans J.A., Bryant S.H.: Cn3D: sequence and structure views for Entrez. *Trends Biochem Sci.* **25**(6) (2000) 300–302
25. Wittenbrink, C.M., Pang, A.T., Lodha, S.K.: Glyphs for Visualizing Uncertainty in Vector Fields. *Proceedings of IEEE Transactions on Visualization and Computer Graphics* **2**(3) (1996) 266–279

Appendix: Geometry Operations and Examples

Operation	Effect	Parameters	# segments created
L	line	L: length (L100) A: angle (A0)	1
T	triangle	h <sup>1</sup>  H: height (h0.5) o O: position of peak along base line (o0.5)	2
C	circle	h H: height (h0.5) o O: perp. offset of center from base line (o0)	2 or 4
Q	quadrilateral	h H: height (h0.5) w W: top width (w1) o O: offset of top segment along base line (o0)	3
D <sub>n</sub>	divide	<i>n</i> : number of segments (1) [l L] <sup>n</sup> : length of each segment ([l1/ <i>n</i> ] <sup>n</sup> )	<i>n</i>
S	shorten	b B: length to shorten from beginning (b0) e E: length to shorten from end (e0)	1
B	add segment going backward		2
Z	zigzag (same as BB)		3
R	reflect	b B: length to shorten reflected geometry from beginning	<i>x</i> <sup>2</sup>
-	none		0

 L_	 LT_	 LTh1o1_	 LC_	 LCh-.5o-.5	 LQ_	 LQh.75w.5o.25
 LD2T_C_	 LD3l.2l.6Qh1_Th-.25_Ch.5o.5	 LBC_D2T_Q	 LZTT_Th-.5o1_Q_D3_Ch-.5o-.5_Se5_Rb5		 LBQh.25w.5o-.25_To-.25_R	

<sup>1</sup> Small caps indicates the value is multiplied by the length of the segment.  
<sup>2</sup> Number of segments reflected, which are all subsequently skipped.

# Negotiating Gestalt: Artistic Expression by Coalition Formation Between Agents

Kaye Mason, Jörg Denzinger, and Sheelagh Carpendale

Department of Computer Science, University of Calgary, Canada  
{katherim, denzinge, sheelagh}@cpsc.ucalgary.ca

**Abstract.** We present a system using semi-autonomous agents to help artists express ideas. Agents control their representation on a canvas via interactions in agent space. They are given rules, which include the ability to form, join, leave and dissolve coalitions. While individually, agents constitute only small parts of the composition, together they form more and more complex parts until the full picture emerges. The artist can take direct control of an agent or coalition if the result is unsatisfactory.

We have implemented the *Surreal* engine that realises these ideas and we present a case study of its usage in producing Mondriaan-like paintings.

## 1 Introduction

In expressing ideas, emotions and opinions, artists often use media that consist of fundamental elements composited into a *gestalt* – a whole that is greater than the sum of the parts. For composers the elemental pieces can be notes, for writers – words, and for painters – strokes on canvas. One way of describing this is to see the style of individual artists as self-defined rules of how elements are grouped together, and an individual composition as created through the development, application and refinement of these rules. In this paper, we consider the implications of this in the medium of computer graphics. A computer graphics image has been taken as a collection of graphical elements, combined through rules provided by an artist working with an algorithm. We take this a step further, and consider these graphical elements as self-interested agents in a multi-agent system. The *gestalt* in our paradigm comes from the *emergent behaviour* of these agents as they seek to follow a set of artist-specified rules.

A practical challenge rests in the sheer numbers of image-elements that can be required to compose an image. One solution has been to write algorithms that deterministically place image-elements on a base image or model, leaving the artist to focus on the creation of this image or model while the algorithm does the image-element placement. The drawback of this approach is the algorithmic dominance of the style of the final composition. While it is acceptable that tools influence a work in any medium, it is important to continue to search for a better balance between artistic expression and algorithmic support.

By applying multi-agent techniques – taking each image-element as a semi-autonomous agent, and the composition as an expression of a multi-agent system – we find a new way of seeing an old problem. Still, it is not immediately

apparent how it is easier for the artist. Not only do artists have to work with large numbers of static elements, but now they have large numbers of agents, each with its own views about what its environment should look like. It is only when we introduce coalition formation that the power of this approach becomes apparent.

When image elements form coalitions, they band together into larger groups that an artist can manipulate, making the tasks of building an image from elements more reasonable. Instead of dealing with masses of tiny elements, the artist can deal with a smaller number of coalitions (groups) of elements. The process of creating an image becomes the process of training agents about the conditions under which coalitions should form, and the properties they should have. Conceptually, the *gestalt* becomes the grand coalition, and the composition is complete when it is achieved to the satisfaction of the artist.

## 2 MAS-Coalitions for Artistic Expression – A Framework

Computers have provided artists not only with a new medium but also a range of possibilities and a detail of control that can be overwhelming. We first look at state-of-the-art computer graphics tools and philosophies behind them. Then we present the framework for our multi-agent based approach.

### 2.1 Artistic Expression Tool Philosophies

Many of the computational tools developed for artists involve an algorithm, which adds elements to a composition to alter its appearance. Strothotte et al. presented a general framework for this process in [12], and since then many different artistic styles have been turned into algorithms, including watercolour painting [3], expressionist, pointillist and impressionist painting [4,5], and pencil sketching [1,2]. All of these techniques have something in common – they involve adding algorithmically placed elements to an artist-defined image or model.

Initiated by Smith [11], researchers are also considering how to best deal with such graphical elements. Smith’s models are defined by a formal grammar, and he uses the term *graftals* to describe his grammar based graphical elements. Smith’s ideas were extended by Kowalski et al. [7]. In these models, Smith’s notion of a grammar-based element is extended to include procedurally-based image elements, which are placed over the final image at points determined to be appropriate by the algorithm. These ideas have been extended by Markosian [9] and by Kaplan [6].

### 2.2 Agents and Coalitions for Artistic Expression

Our multi-agent framework for aiding artistic expression follows in the tradition of graftals, but extends it to give a picture element a more active role, even to making it nearly autonomous. Picture elements pursue goal fulfillment and the final composition or *gestalt* results from emergent agent behaviour. In the

following, our framework is described in terms of agents, the agent space, and the emergent gestalt resulting from coalition formation.

**The Agents:** Within our framework, an agent  $Ag = (h, R, S, T)$  consists of a happiness function  $h$ , a set of rules  $R$  governing its behaviour, a set  $S$  that contains the agent's possible subjective spatial understandings and a set  $T$  with the possible values of the agent's internal data tables. The function  $h$  is selected by the artist to express what makes the agent happy. It is the goal of each agent to optimize its individual happiness. The set of rules  $R$  consists of behavioural rules  $R_{Beh}$  and rules  $R_{Exp}$  that guide how the agent expresses itself on the canvas ( $R = R_{Beh} \cup R_{Exp}$ ). The agent selects the behavioural rule that will optimise its happiness by evaluating the possible resulting elements from  $S$ . If no rule increases happiness, then no behavioural action is taken. The rules in  $R_{Exp}$  deal with how the agent manifests on the canvas. All the rules in  $R_{Exp}$  that can be applied will be applied in addition to the selected rule from  $R_{Beh}$ .

**The Agent Space:** Agents exist in a space that is full of other agents each striving for their own idea of what constitutes happiness. However, in this graphics application the term *space* is significant. The space in which we view images we will call *canvas space*. Our conceptual framework of agents that make decisions about their own expression requires an additional spatial abstraction. These agents do not exist in the canvas space, but in *agent space*. From agent space, it must be possible for the agents to create a representation in canvas space. The set  $S$  of possible spatial understandings of an agent represents the agent's subjective view of both spaces.

Since artists often use and reuse picture elements in different contexts and the same picture element may occur thousands of times in a given image, it should be possible to define an agent for a particular picture element once. To support this, we introduce the concept of an agent *tribe*. Agents of a tribe start with the same  $h$ -function, the same sets  $R$ ,  $S$ , and  $T$ , but they can start with different actual values for  $S$  and  $T$ .

**Coalitions and the Emergent Gestalt:** Coalitions form when an agent discovers that the fulfillment of its goals is compatible with goals of other agents. Our coalitions, in common with other MAS (see, for example [13]), operate like individual agents,  $Co = (h, R, S, T)$ . Rich variation in coalition formation is possible and necessary in our application to support expression.  $Co$  can be arrived at through: construction from the corresponding components of the member agents, influence from a new agent member, or use of a predefined set. Due to the artist's ability to plan and interact, we can allow coalitions to become members in other coalitions, and agents and coalitions can even be members of several coalitions.

In addition to the types of rules agents have, coalitions also have rules that deal with accepting additional agents and with dissolving a coalition. Individual agents can leave coalitions; however, if an agent selects to leave, it can make sense to dissolve a coalition. For example, when a coalition contains only one other agent or when the artist finds a coalition unsatisfactory. This requires that such rules are treated specially and they are executed without considering happiness.

### 3 The *Surreal* Engine

The artist is an important aspect of our system. If our agents behave badly, interference is possible (and should take place). This solves some issues, but also presents challenges. While the principles of our framework apply equally to writing or to music, the *Surreal* engine is presently aimed at visual arts. Therefore we must acknowledge and respect the skills of visual artists. These rely largely on visual feedback, and we must provide that feedback in reasonable time, and without overwhelming detail. We call this the Visual Principle.

#### 3.1 Defining Agent Space

Our agent space is topologically equivalent to a hyper-torus — a cube with periodic boundary conditions. This decision was made for simplicity, to allow us to easily use existing techniques in MAS research which operate on that basis. If an agent leaves one face of the hyper-torus, it reappears at an opposite position. This simplifies agent movement — the agents cannot move outside valid bounds — they just move and deal with their new environment.

Given the large number of agents involved, it can be hard for one agent to find others. If each agent has to compare its position with all others to find neighbours, computation time is  $\mathcal{O}(n^2)$ . This inhibits prompt feedback. To address this, we introduce a coordination artifact after Omicini et al. in [10] – a dynamic spatial grid in agent space. We use a grid, as opposed to other methods of dividing space (octrees, BSP-trees, etc), because we have no guarantees of continuity in the density of agent space, and the grid does not require any maintenance as this shifts over simulation steps. That said, it is possible that in future work, this function could be offered in a better way.

When an agent is placed or moves, it registers with the grid. Each grid cell keeps a list of registered agents in its territory. An agent can query its own cell or neighbouring cells for a list of registered agents. We allow the artist to set the grid resolution dynamically if desired, to address agent density issues, and to prevent the algorithmic degeneracy that occurs if all agents are in the same grid cell.

#### 3.2 Defining Agents

As already discussed, we use tribes to define similar agents with a prototypical agent. Here, we consider prototypical agent definitions in four parts, based on the tuple which defines an agent in our framework. For each, we look at specifics of our implementation. For ease of consideration of practical issues, we rearrange the discussion slightly from the last section.

**Agent Spatial Understanding (*S*):** All agents in the *Surreal* engine begin with no neighbourhood understanding. They know that they exist and they understand the nature of agent space. They know that there may or may not be other agents, and they know how to perform searches for these agents, and

how to move themselves in agent space. Using this, an agent can find others and share information, including tribal membership, data tables, and even spatial understanding. In general, agents have the ability to see any other agent. It is possible, however, to exclude an agent from noticing agents with particular properties. Likewise, it is possible for an agent to choose not to register with its grid cell, making it invisible to other agents performing grid searches.

**Data Tables ( $T$ ):** Data tables can hold any information the artist wants the agent to have, within the limits of system memory. The tables can be referenced by the agent's rules in  $R$  (see below). This data may or may not be freely visible, depending on the simulation. Some examples of data which may be stored are:

- The agent's position in agent space, as  $(p = (x, y, z, w))$ .
- Visual representations for the agent as images, models, or drawing methods.
- Transformation  $M = M_{Translate}(t_x, t_y, t_z) \cdot M_{Rotate}(\rho, \phi, \theta) \cdot M_{Scale}(s_x, s_y, s_z)$ .
- Critical values for an agent's happiness  $(\alpha_0 - \alpha_n)$  in the range  $0 \leq \alpha_i \leq 100$ .
- Special data that is held by the agent for the particular simulation.

Generally, data tables are built up as a simulation runs – forcing the artist to deal with it can be overwhelming, a violation of the Visual Principle. However, it is possible for the artist to interfere, and set or adjust values if they wish to get that deeply involved.

**Agent Happiness ( $h$ ):** We represent the agent's happiness as a percentage. No happiness is 0%, while a state of perfect happiness is 100%. Depending on the simulation, an agent may begin with no happiness, with perfect happiness, or with something in between. Every simulation step, each agent adjusts its happiness by checking for relevant factors. In many simulations, it is useful to decrement each agent's happiness by a small amount in every simulation step. This forces the agents to continue to actively improve their situation.

**Rules ( $R$ ):** The *Surreal* engine supports the expression and behaviour rules outlined in the last section as follows:

*Expression Rules* use information from an agent's data tables ( $T$ ) to represent that agent in *canvas space*. No conflict resolution is applied — in every simulation step, each display rule that can be executed is executed. This may result in the agent having no representation or multiple representations.

- *Universal Rules* are always applied. They include a reference to a visual representation, a transformation matrix, and possibly other items from the agent's data tables (see above).
- *Conditional Rules* are rules of the form  $A \rightarrow B$ , where  $A$  is a condition that can be tested, and  $B$  is an expression rule, formed like a universal rule. The predicate  $A$  could be a conjunction, formed as  $A = a_0 \wedge a_1 \wedge \dots \wedge a_n$   $a_0, \dots, a_n$  are some predicates defined on either agent or canvas space and tested on the agent's subjective understanding.

*Behaviour Rules* can rely on information in the agent's spatial understanding ( $S$ ), as well as in the agent's data tables, as follows:

- *Exploration Rules* involve the agent gathering data and exploring agent space. These rules often modify the agent’s spatial understanding ( $S$ ) either by gathering new information or by choosing to forget information.
- *Movement Rules* change the position of the agent in agent space. We represent this in homogeneous coordinates (a position tuple  $p = [x, y, z, w]$ ), stored in the agent’s data tables. Movement rules alter these values. The agent can also report its positional change to the agent space grid, so its registration can be updated. Agents that are deliberately hiding from other agents would not do the latter.
- *Relationship Rules* alter an agent’s understanding of its relationships with other agents. These can include rules involving the formation and breakup of coalitions, in requesting information from another agent, or in answering requests for information.

### 3.3 Defining Coalitions

When defining and forming coalitions, we use an encounter-based approach, like Lerman [8]. Because there is no agent with an overall view, agent decision making is performed in a bottom-up manner. That said, the agent’s understanding of its own goals is heavily influenced by the artist. The artist may provide rules which instruct the agent that it would be well-served to form a coalition with agents with particular properties. Still, an agent’s decisions are always based on what information it has collected. We have generally opted to use one agent coalition’s director. While it is possible to use voting schemes or other methods within the *Surreal* engine, these can consume resources and prevent us providing the prompt feedback required by the Visual Principle.

Because we allow an agent to belong to as many coalitions as it finds advantageous, conflicts arise. In practice, this does not happen often, because the simulation specifications are planned with care. However, when conflicts do arise, an agent must leave one of the conflicting coalitions. Since coalitions, once formed, can represent themselves as an agent in the system, we define them to have the same components as an agent. More formally, we define a coalition as a tuple  $Co = (h, R, S, T)$ , as follows:

**Coalition Spatial Understanding ( $S$ ):** A coalition begins with the spatial understanding of all its members. This allows the coalition to make decisions based on information that is at least as good as its members would have on their own. If a coalition  $Co = \{Ag_0, Ag_1, \dots, Ag_n\}$  then  $S_{Co} = \{S_{Ag_0} \cup S_{Ag_1} \cup \dots \cup S_{Ag_n}\}$ .

Since we use directed coalitions, all agents share their understanding  $S_{Ag}$  with the director. This information is left in the repositories of the individual agents. Copying the data to a different location takes time, which prevents the system from offering adequate visual feedback. When an agent leaves a coalition, it takes its spatial understanding  $S_{Ag}$  with it, and when a new agent joins, its understanding is automatically added to  $S_{Co}$ .

**Coalition Data Tables ( $T$ ):** The director of the coalition has access to the data tables of all members. Again, if a coalition is  $Co = \{Ag_0, Ag_1, \dots, Ag_n\}$ ,



then  $T_{Co} = \{T_{Ag_0} \cup T_{Ag_1} \cup \dots \cup T_{Ag_n}\}$ . Special data items may be created for the coalition — for example, the coalition’s current happiness ( $h_t$ ).

**Coalition Happiness:** Coalition happiness is similar to agent happiness. When the coalition comes into existence, its happiness measure is created by the coalition’s director, as distinct from that agent’s own happiness. No bottom-up approach to coalition formation can ensure optimal or even desirable coalitions. We must allow agents the possibility of leaving or dissolving a coalition. Though agents must follow coalition rules, and can gain and lose happiness through the coalition’s actions, they still keep track of their own happiness. If it falls below one of their critical values, an agent will leave the coalition.

A special case of leaving occurs when the happiness of the director, or the happiness of the coalition falls below a critical value. In either case, the director may decide that the coalition is failing, and that it should dissolve. This is similar to Weiß [13]. If this happens, any special data stored by the director for the coalition, like coalition happiness and critical values, is deleted.

**Coalition Rules ( $R$ ):** Evaluation of the rules is done by the assigned director that also initiates the execution of the rules.

*Expression Rules* for coalitions are acceptable in the *Surreal Engine*. They are accomplished in much the same way as for individual agents, and are orchestrated by the director of the coalition. Usually having them means that individual agents only use conditional rules for individual expression, with the condition being an absence of coalition membership.

*Behaviour Rules* can be divided into movement, exploration, and relationship rules, as with agents. The form of these rules will depend on the simulation, however, in our simulations, we have found some generalities are useful. While in a coalition, members generally agree to give up any actions involving independent movement, unless instructed by the coalition’s director. Exploration by individual agents can continue as before, and honest agents share new information with a coalition’s director, although cheating agents may not. Agents in a coalition can continue to form relationships with agents outside, and even join multiple coalitions. If there is a conflict in agreements, they will leave whichever coalition has been giving them the least happiness.

## 4 Session – Piet Mondriaan

To illustrate the workings of our system we have chosen a simulation that imitates the style of the artist Piet Mondriaan. The inherently mathematical aspects of Mondriaan’s abstract compositions allow us to create a straight forward set of rules for demonstrative purposes.

It should be noted that instead of trying to create our own artistic style, we are actually attempting to imitate the style of an existing artist, as a proof of concept. We have a number of original simulations with this system, based on our own ideas and those of other artists, but the simplicity here allows us to focus on the system, and not the details of the simulation:

## 4.1 Setup and Definitions

In order to set up the fundamentals for Mondriaan, the artist has specified four tribes of agents, each of which contains a piece of information required to draw a primitive to canvas space, and three specialised coalitions. All of these have some common elements:

- *Happiness Function.* The happiness of all agents and coalitions is defined in the range  $0 \leq x \leq 100$ , the lower bound being no happiness, and the upper perfect happiness. The happiness of the agent in the current time step ( $h_t$ ) is calculated by  $h_t = \min(\max(h_{t-1} - 1, 0) + h_{act}, 100)$ , where  $h_{t-1}$  represents the happiness at the last time step, and  $h_{act}$  represents happiness gained by particular actions. Some values for happiness increases could be: joining a coalition: 50; getting a new member: 25; showing off on canvas: 1.
- *Despair Condition.* We define critical values  $\alpha, \beta, \gamma, \dots$  for each agent and coalition (such that  $\alpha \leq \beta \leq \gamma$ ), beyond which an agent will take particular actions. The lowest of these ( $\alpha$ ) is a special value, below which an agent will enter a state of despair. In this state, it will accept any coalition offer it gets. In this way, we encourage agents to join the gestalt grand coalition, even if they have not been able to enter the intermediate coalitions.
- *Distribution Procedure.* At the start, agents of all four tribes are distributed randomly across the canvas in numbers that have been previously specified by the artist.

**Orientation Tribe:** Agents in this tribe provide later line coalitions with an orientation, i.e., horizontal or vertical (in  $T$ ). The two orientations are approximately evenly distributed across agents. In addition to holding this information, orientation agents act as directors of line coalitions. They have no expression rules, and six behaviour rules. These rules move the agent in agent space until a coalition possibility is found, form line coalitions and announce this formation. They record other line coalitions (in  $S$ ) to track the next drawing location. They may also reach a despair condition, as described above.

**Width Tribe:** These agents provide line coalitions with line width information (in  $T$ ). The width values are distributed across agents such that there are few lines at the widest width, and more at the narrowest. They have no expression rules and four behavioural rules which direct them to move in agent space until they are invited into a line coalition, or reach their despair condition.

**Offset Tribe:** Offset agents provide line coalitions with information about how far to offset from the last line drawn of the same orientation (in  $T$ ). This allows the creation of wide, unused portions of canvas, as favored by Mondriaan. These agents have no expression rules, and four behaviour rules which provide the same functionality as the width tribe's rules.

**Colour Tribe:** These agents provide fill coalitions with a colour (in  $T$ ). They also direct fill coalitions, and are responsible for recruiting members. They have

no expression rules, and five behaviour rules, which involve movement in agent space, recruitment of line coalitions, and the despair condition.

**Line Coalitions:** A complete line coalition consists of an orientation (director), a width and an offset agent. They have one universal expression rule – to express as a line with a width specified by the width agent, an orientation specified by the orientation agent, and offset from the last line drawn of the same orientation, or the edge, in the absence of any line. They have four behaviour rules, moving in agent space until invited to join a fill coalition by a colour agent, and a despair condition. A line coalition may join multiple coalitions, as long as no two of these coalitions result in adjacent fill areas.

**Fill Coalitions:** A complete fill coalition consists a colour agent (director), and two line coalitions of each orientation. The coalition’s expression is then a rectangular fill of the area between these four lines. This expression may overlap other line coalitions. If so, it is drawn on top of them. The fill coalition has five rules, related to movement and exploration in agent space, the formation of gestalt coalitions, and the despair condition.

**Gestalt Coalitions:** A gestalt coalition is the final attempt of agents and coalitions to stabilise their environment. It is directed by the first agent or coalition to propose its formation. Complete fill coalitions will always accept a proposal to join this coalition. Other agents which have reached their despair condition will always accept a proposal to join. Another gestalt coalition will always accept a proposal to join, and give control to the director that made the request. Gestalt coalitions use the happiness function  $h_t$  modified by the fact that their existence allocates one happiness percentage to each member in each time step. They have no expression rules, and no behaviour rules – the rules for recruiting new members are contained within the rules for a fill coalition.

## 4.2 Execution

We will not show all steps in the execution, only check points at which something crucial occurs. We also do not show artist interaction – human computer interaction is a complex issue for this application, as with any application, and the primary concern of this paper is with the MAS aspects.

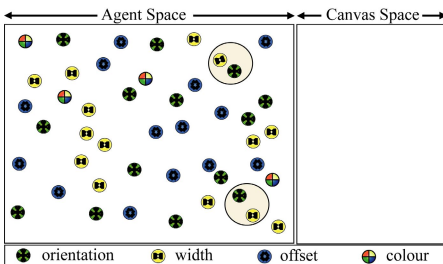


Fig. 1. First coalitions appear

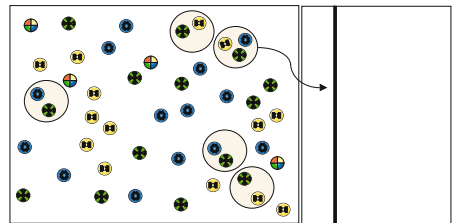
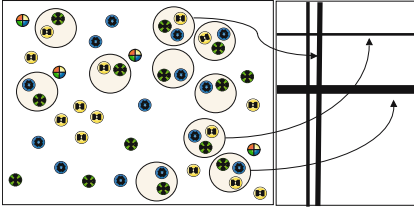
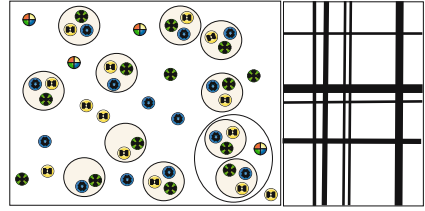


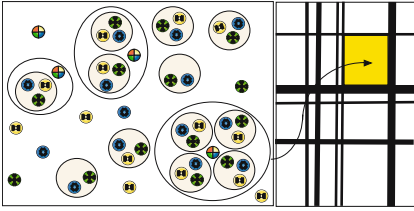
Fig. 2. First line coalition



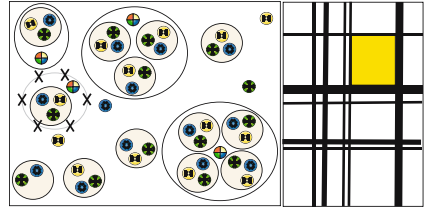
**Fig. 3.** More line coalitions



**Fig. 4.** Budding fill coalition



**Fig. 5.** First fill coalition



**Fig. 6.** Fill coalition dissolves

At the start of the simulation, agents are scattered over agent space, unaware of their surroundings. They begin to come together in coalitions, but none yet has the requisites to express itself in canvas space. See Figure 1. Figure 1 also presents the legend for all the figures.

The first coalition of three agents from the right tribes appears, and a corresponding line is drawn in canvas space with the width, offset and orientation specified by the agents in the new coalition. See Figure 2.

More coalitions meet the requisites, and more lines are drawn in canvas space with the offsets, widths and orientations specified by their members. See Figure 3.

Coalitions of line coalitions begin to form. At this point, however, no line contains the requisite pair of coalitions of each orientation, so no fills can be drawn. See Figure 4.

The first fill coalition appears when four line coalitions find one another. There are two horizontal and two vertical line elements, and on the canvas the fill is drawn between the lines represented by these elements. See Figure 5. Fill coalitions continue to form. At this point, the colour agent directing one fill coalition decides the coalition is not beneficial, and dissolves it. See Figure 6.

New fill coalitions appear, including fill coalitions sharing members. Two coalitions which share one line coalition in agent space will share one line in canvas space. See Figure 7. The coalitions of fills begin to ally with one another and with free agents and line coalitions. Together, they begin to form the gestalt coalition. Once in this coalition, the happiness of each agent will grow to perfect. See Figure 8.

The grand coalition is achieved. At this point, the composition is complete, unless the artist wishes to interfere with it, to break coalitions, or to make changes to individual agents. See Figure 9.

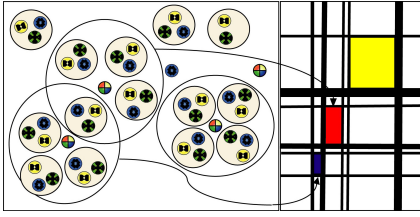


Fig. 7. More fill coalitions

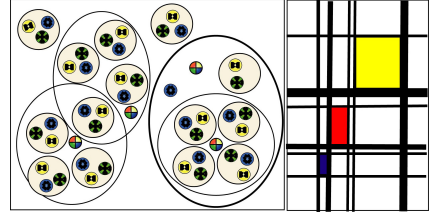


Fig. 8. Gestalt coalition begins

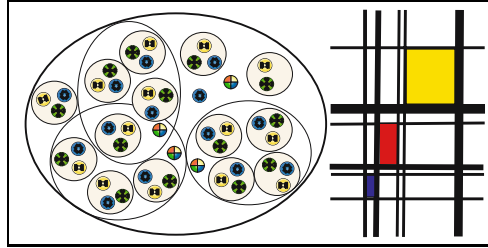


Fig. 9. Gestalt grand coalition

Note that every time we run *Surreal* with these agents, we get a different composition, because the initial placement and data distribution across agents is randomised. As a result, the coalitions develop differently.

## 5 Conclusions and Future Work

We have presented a framework for the application of multi-agent systems and coalition formation to the process of creating artwork. We have shown how the *Surreal* engine uses this framework, using a simple case example – defining agents and coalition types for art similar in style to the works of Piet Mondriaan.

The basic concepts we propose allow for various instantiations and while *Surreal* already offers some choices, more of them will have to be added to cater to more styles. Figure 10 shows an image in the Japanese *seigaiha* style created with *Surreal*. The use of these techniques in the areas of music, writing, and

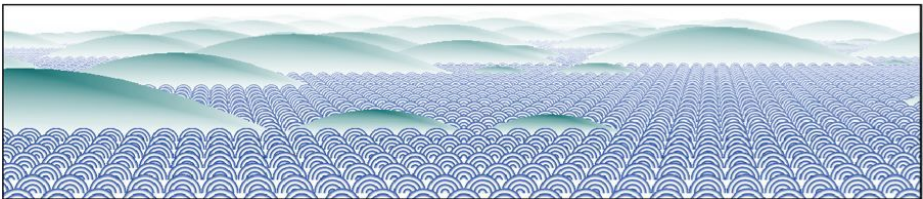


Fig. 10. Seigaiha pattern done with agents using this framework

other arts remains to be developed, but we believe that the same techniques we have used could be applied in those problem domains, and we look forward to tackling these areas in the future.

## Acknowledgments

Research supported by the Natural Sciences and Engineering Research Council of Canada and the Alberta Informatics Circle of Research Excellence.

## References

1. M. Costa Sousa and J. W. Buchanan. Computer-generated graphite pencil rendering of 3d polygonal models. In *Eurographics*, 1999, pp 199–207.
2. M. Costa Sousa and J. W. Buchanan. Observational model of blenders and erasers in computer-generated pencil rendering. In *GI*, 1999, pp 157–166.
3. C. J. Curtis, S. E. Anderson, J. E. Seims, K. W. Fleisher, and D. H. Salesin. Computer-generated watercolor. In *SIGGRAPH*, 1997, pp 421–430.
4. A. Hertzman. Painterly rendering with curved brush strokes of multiple sizes. In *SIGGRAPH*, 1998, pp 453–460.
5. A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. . Salesin. Image analogies. In *SIGGRAPH*, 2001, pp 327–340.
6. M. Kaplan, B. Gooch, and E. Cohen. Interactive artistic rendering. In *NPAR*, 2000, pp 67–74.
7. M. A. Kowalski, L. Markosian, J. Northrup, L. Bourdev, R. Barzel, L. S. Holden, and J. F. Hughes. Art-based rendering of fur, grass, and trees. In *SIGGRAPH*, 1999, pp 433–438.
8. K. Lermann and O. Shehory. Coalition formation for large-scale electronic markets. In *ICMAS*, 2000, pp 167–174.
9. L. Markosian, B. J. Meier, M. A. Kowalski, L. S. Holden, J. Northrup, and J. F. Hughes. Art-based rendering with continuous levels of detail. In *NPAR*, 2000, pp 59–66.
10. A. Omicini, A. Ricci, M. Viroli, C. Castelfranchi and L. Tummolini. Coordination Artifacts: Environment-Based Coordination for Intelligent Agents. In *AAMAS*, 2004, pp 286–293.
11. A. R. Smith. Plants, fractals, and formal languages. In *SIGGRAPH*, 1984, pp 1–10.
12. T. Strothotte, B. Preim, A. Raab, J. S. Mann, and D. R. Forsey. How to render frames and influence people. In *Computer Graphics Forum* 13, 1994, pp 455–466.
13. G. Weiß. *Distributed Machine Learning*. Infix-Verlag, Sankt Augustin, 1995.

# Metrics for Functional and Aesthetic Label Layouts

Knut Hartmann, Timo Götzelmann, Kamran Ali, and Thomas Strothotte

Department of Simulation and Graphics,  
Otto-von-Guericke University of Magdeburg,  
Universitätsplatz 2, D-39106 Magdeburg, Germany  
{knut, timo, kamran, tstr}@isg.cs.uni-magdeburg.de

**Abstract.** Co-referential relations between textual and visual elements in illustrations can be encoded efficiently through textual labels. The labels support students to learn unknown terms and focus their attention on important aspects of the illustration; while a functional and aesthetic label layout aims at guaranteeing the readability of text strokes as well as preventing the referential mismatches.

By analyzing a corpus of complex label layouts in hand-drawn illustrations, a classification of label layout styles and several metrics for functional requirements and aesthetic attributes were extracted. As the choice of a specific layout style seems largely determined by individual preferences, a real-time layout algorithm for internal and external labels balances conflicting user-specific requirements, functional and aesthetic attributes.

## 1 Introduction

Almost all illustrations in scientific or technical documents employ a large number of textual annotations in the form of labels. The labeling provides a direct way to visualize the *co-referential* relations between textual and visual elements.

Textual *labels* either overlay visual objects or are placed outside (*internal* vs. *external* labels). Moreover, the form and orientation of internal labels can target on readability (axis-aligned typing, see Fig. 1-1) or convey topological information (the text stroke provides indication for the shape and extent of area and line features, see Fig. 1-2) [14]. For external labels (see Fig. 1-3) additional meta-graphical objects like *connecting lines* and *anchor points* establish a co-referential relation between labels and visual objects. However, sometimes artists prefer to omit both anchor points and/or connecting lines if the layout remains unambiguous.

Learning materials typically employ annotated illustrations in order to convey many unknown terms in a domain-specific or foreign language in parallel, whereas complex spatial configurations are often described textually. However, an interactive exploration of complicated subjects (e.g., technical devices or organic structures) through 3D browsers can increase the learning efficiency [22,20].

The majority of learning and instructional material still employs static illustrations, as the depicted objects are carefully drawn by hand to convey the intended function and due to the complexity of the label layout problem. But in many technical and scientific domains the required resources are already available and can be exploited in on-line tutoring applications. Moreover, several research groups in computer graphics focus

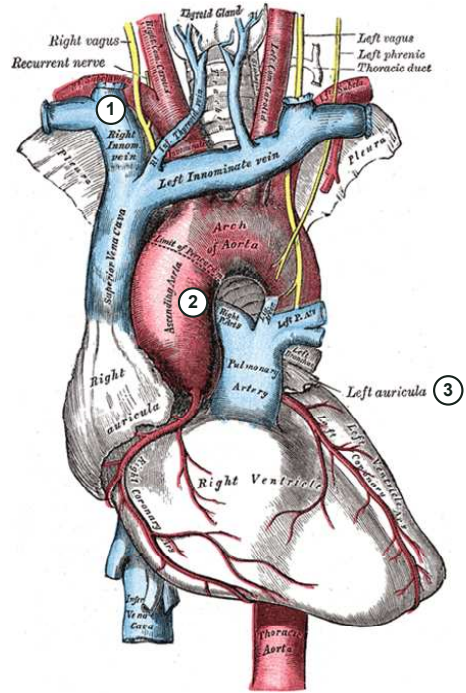
on those illustration techniques which are frequently used in technical and scientific illustrations (e.g., transparency [8], cutaways [8], explosion diagrams [1]). However, the automatic labeling of 3D models received almost no attention from the industrial or research community. Therefore, we implemented several layout styles for internal and external labels and integrated real-time label layout algorithms into an interactive 3D browser. Thus, the presentation style as well as the label content can be adapted in order to meet specific individual preferences and learning tasks.

An automated label layout system (a) *selects* those labels which the layout should contain, (b) *classifies* them into internal or external labels, and (c) *determines* all *style-specific parameters* for those labels.

This paper is organized as follows. Sec. 2 presents the State-of-the-Art of automatic labeling systems. From a manual analysis of hand-made illustrations, we extract-ed several label layout styles and classified them according to common properties. These individual layout styles aim at achieving specific functional requirements or are associated with a set of aesthetic criteria (Sec. 3). The metrics to measure these functional requirements and aesthetic attributes are presented in Sec. 4. Users can adjust the impact of these metrics with weights. These uniform metrics for internal and external labels allowed us to integrate labeling algorithms in a common layout architecture (see Sec. 5). After presenting some examples in Sec. 6, we discuss directions of future research (Sec. 7). Finally, Sec. 8 summarizes the contributions of our work.

## 2 Related Work

The efficiency of multi-modal information presentation was studied within psychology, aiming at extracting principles that guarantee an effective design. MAYER [17] imposed the *spatial continuity principle*, assuming that the cognitive load to integrate co-referring multi-modal expression depends on the spatial distance between related elements. Hence, annotations should be placed as near as possible to their co-referring visual object. This distance is minimal, if textual annotations are drawn as overlay. Moreover, the *readability* of textual labels have to be considered.



**Fig. 1.** Illustration with internal (1 and 2) and external labels (3). (Source: [13])



These principles also dominate the label layout for point, line, and area features in cartography. There are numerous approaches to translate IMHOF's informal principles of label placements [14] into numeric scores or constraints which are solved using numerical optimization methods [7,10]. Several research groups also focus on interactive cartography (e.g., to develop algorithms for real-time label layout of dynamic maps ([18,9]) or to consider user-specific requirements within the map generation [2]. Recently, layout algorithms for external labels extend the classical cartographic methods [5].

These techniques influence label layout algorithms for internal and external labels in computer graphics and information visualization [11]. Recently, in AR/VR the term *view management* is used for a more general, but related problem: the smooth integration of additional 2D information (images, texts, annotations) into the view plane [6,4]. Other researches employ these techniques for the generation of instructional [15] or tutoring materials [21,23].

However, these research prototypes implement a small subset of those label layout styles found in scientific or technical documents [21] and they are based on rough shape approximations [6] or rely on user interaction to achieve an appealing label layout [15,23]. The approach of ALI et.al. [3] first implements a set of layout styles for external labels in an interactive 3D browser which also considers the correct shape of complex 3D objects.

### 3 Label Layout Styles

The material in this section is based on a manual analysis of label layouts in hand-drawn illustrations. For this purpose, we chose anatomic atlases, anatomic textbooks, and visual dictionaries as their illustrations employ very elaborated label layouts of extraordinary quality.

The manual analysis reveals that human illustrators use a number of different label layout styles with style-specific illustration techniques and properties. As their selection seems to be highly determined by aesthetic preferences, users should be able to specify style preferences and priorities for functional requirements.

There are two styles for **internal labels**: they could either be aligned to a horizontal line or their curvature can provide indication of the shape and extent of area and line features. Fig. 1 contains both types of internal labels. Moreover, slanted straight text strokes are used. As these strategies aim to achieve two conflicting cartographic principles (legibility vs. clear graphic association [14, pg.129]) users have to specify appropriate priorities.<sup>1</sup> The layout algorithm has to also consider additional constraints such as the amount of available space to display internal labels for linear and area features and the available background space for external labels.

We extracted three main *features* of the layout styles for **external labels**:

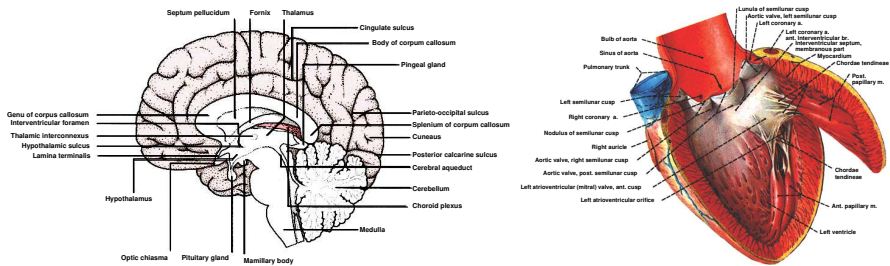
1. the *style of connecting lines* (orthogonal with bends vs. straight lines),
2. the *labeling area* (rectangular arrangement vs. circular arrangement), and
3. the *label alignment* (labeling regions vs. object shape).

<sup>1</sup> In Sec. 4 we introduce the terms *unambiguity* and *readability* as synonyms to IMHOF's cartographic terms.

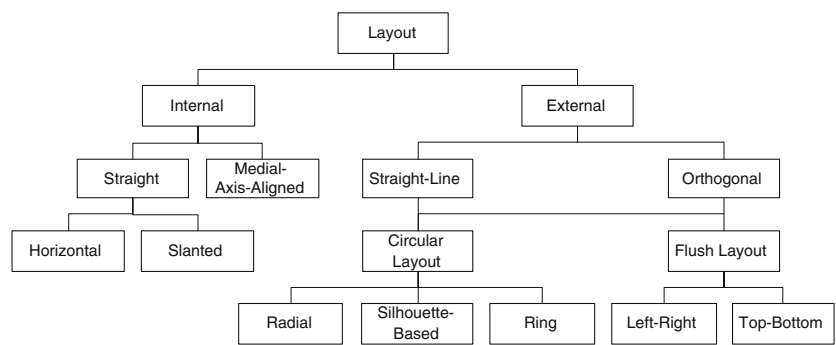
These features are used to define layout styles for external labels and their individual properties. The label layout of Fig. 2-left can be characterized as: orthogonal-lined, rectangular labeling area, and label-aligned, whereas the layout of Fig. 2-right is best described as: straight-lined, circular labeling area, and shape-aligned. For the sake of brevity, we refer to these labeling styles respectively as *flush left-right* and silhouette-based circular layouts according to the hierarchy presented in Fig. 3.

There are several functional requirements which are common to all layout styles:

- 1. Place labels near to their corresponding objects,
- 2. Labels must neither overlap one another, nor the objects, and
- 3. Prevent crossings of connecting lines.



**Fig. 2.** External label layout styles: Orthogonal (left) and circular layouts (right). (Source: [24, pg.317] and [26, pg.81]).



**Fig. 3.** Layout classification

However, these functional requirements can interfere with style-specific aesthetic criteria. The average length of connecting lines for layout styles which utilize a separate labeling area and a label-alignment (i.e., *flush*) is greater than for styles which use a circular label arrangement and object alignment. Moreover, layout styles can incorporate additional aesthetic criteria. An important aesthetic criterion for *orthogonal*

layouts, for example, is to minimize the number of bends in orthogonal, axis-aligned connecting lines. Finally, functional requirements as well as aesthetic criteria might be not completely satisfied. For example, note that two connecting lines are crossing on the left side of Fig. 2-left. Moreover, the distribution of labels is not balanced on the right label area of Fig. 2-right. These examples demonstrate the complexity of the label layout problem.

Our system implements layout styles as classes and enables the illustrators to specify their own requirements on labeling styles.

## 4 Metrics of Functional Requirements and Aesthetic Attributes

The main challenge for human illustrators while placing labels is to consider a number of conflicting functional requirements and aesthetic attributes such as readability, unambiguity, aesthetic considerations, media capabilities, publishing costs, and subjective preferences. The label layout in interactive 3D browsers introduces additional requirements for an aesthetic *dynamic* label layout: frame-coherency and efficiency. Finally, the label layout should be *adaptive*, i.e., it should reflect contextual requirements. This could be achieved by considering the interaction context and user-specific requirements within the label selection and by displaying dynamic contents within the labels.

An aesthetic label layout (*form*) balances these contradicting requirements and supports the intended *function*<sup>2</sup> (i.e., explain the visual subject through annotations). There are different aspects within these criteria which have to be considered for internal and external labels:

The layout **readability** is affected by the label placement as well as from font attributes:

Internal Labels	Area and linear features have to provide enough space to contain internal labels, otherwise they are considered as point features and labeled externally. The text strokes could be either horizontally aligned or they should be drawn as smooth as possible, as a high <i>curvature</i> reduces text readability. <i>Steep</i> text strokes should be avoided as well. If the <i>path length</i> exceeds the required length to display text strokes, then search an appropriate path segment. Moreover, a minimal contrast between letterings and its local environment has to be guaranteed.
External Labels	External labels should neither overlap one another, nor the connecting lines and visual objects. As the majority of tutoring applications use a uniformly colored background, a maximal contrast for the label text can easily be achieved. A minimal contrast between meta-graphical objects (connecting lines and anchor points) and its local environment has to be guaranteed.

**Unambiguity.** The layout should guarantee that the co-referential relation between labels and their associated visual objects can be easily recognized and prevent referential mismatches:

<sup>2</sup> The famous dictum of the architect Louis H. SULLIVAN “form ever follows function” became one of the most influencing guidelines in industrial design due to MIES VAN DER ROHE and other artists from the Bauhaus school.

Internal Labels	Text strokes should be placed over salient regions of line or area features and must not leave the object's interior. Moreover, text strokes should not be placed at very narrow areas.
External Labels	Labels should be placed as close as possible to their co-referential visual objects. Anchor points must overlay their corresponding visual objects and be line-connected to related labels. The number of bends in connecting lines should be minimized and anchor points should not form clusters.

**Aesthetic Considerations.** Aim at achieving a symmetric layout and prevent visual clutter:

Internal Labels	If an internal label cannot be projected in an aesthetic way (e.g., high curvature) it should not be displayed internally but externally. However, this decision should consider a user-defined threshold value.
External Labels	The distribution of labels and anchor points in the layout should be neither too scattered nor too uniform. Moreover, external labels should either be aligned with respect to one another (horizontal or vertical alignment) or along the silhouette of visual objects.

**Frame-Coherency.** To prevent visual discontinuities during user interactions the distance of identical layout elements between subsequent frames have to be minimized:

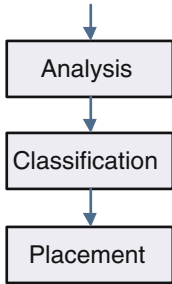
Internal Labels	Even minor changes in the shape of objects may result in drastic changes of the skeleton. Therefore, those segments of the skeleton which minimize the displacement of text strokes are extracted. However, this feature should be disabled to achieve more aesthetic text strokes for static images. Moreover, <i>salience</i> (i.e., larger area features or longer line features) offers a greater stability in the layout.
External Labels	The displacement between subsequent anchor points and label positions should be minimized. Moreover, the number of permutations in the label sequence and the assignment of labels to the different label area should be minimized.

Fig. 5 summarizes the metrics. They can be used to evaluate the quality of a label layout with respect to functional requirements and aesthetic criteria. However, the user should be able to adjust the impact of individual criteria. Moreover, we store these values in log-files while the user interacts with a 3D browser in order to analyze the overall aesthetic measure of different parameter configurations and layout styles.

## 5 Label Layout System

The computation of label layout can be considered as an optimization problem, as the placement of an individual label might effect the quality of the label layout in general. Thus, even a simpler problem (finding an optimal layout for the point-feature labeling problem) has been proven to be NP-hard [16]. Therefore, our layout algorithm heavily relies on several layout heuristics.

First, we estimate the quality of internal and external placements of labels locally (*analysis*). Since the layout algorithms for both label classes are based on medial axis transformations, a common data structure, the skeleton graph, is also constructed in this moduleScene



**Fig. 4.** Label layout architecture

Due to the spatial continuity principle (see Sec. 2) the classification algorithm prefers to label internally, provided there is enough space and the readability has some defined quality. However, an internal labeling is enforced if there is not enough space to display external labels.

The *placement* module determines the remaining parameters required for the specific layout style. Internal labeling selects an appropriate path segment on the object skeleton to display the original or abbreviated text. For external labeling, a point on the skeleton is chosen as anchor point which optimally satisfies the set of functional and aesthetic heuristics. In order to enhance the readability and consistency, the label text is rendered with identical font parameters for internal and

external labels. Moreover, we consider the local contrast around internal letterings to enhance the contrast with halos. The architecture of the label layout system is presented in Fig. 4.

Attribute	Metric	
	Internal	External
<b>Readability</b>	Curvature	No. of label - label overlaps
	Steepness	No. of label - connecting line overlaps
	Contrast	
<b>Unambiguity</b>	Saliency	Avg. length of connecting lines
	Min. Extend	No. of bends in connecting lines
		Saliency of anchor points
		Anchor point clusters
<b>Aesthetic Considerations</b>	Curvature	No. of connecting line intersections
		Label distribution
		Anchor point distribution
<b>Frame-Coherency</b>	Label displacement	Anchor point displacement
	Saliency	Label displacement
		Label permutation

**Fig. 5.** Functional and aesthetic metrics

The metrics of functional and aesthetic attributes presented in Sec. 4 are subsequently used to improve an initial label layout through efficient real-time optimization processes. A detailed description of our implementation is provided in two technical papers [12,3].

## 6 Results

Fig. 6 and 7 present screen-shots of our integrated label layout system [12] and the specialized layout system for external labels [3]. The main advantage of the integrated system is that it can adapt its presentation style (internal vs. external) according to the available space and the user preferences. Fig. 6 demonstrates this feature by zooming into the motor model.

The interactive exploration of 3D models also improves the annotation of geometric models with correct references by the domain expert. The domain expert can interactively specify the content to be displayed in labels. Moreover, abbreviations can be specified, which are displayed in internal labels. These results are stored in an internal database and reused in subsequent interactions with these 3D models. For unknown geometric objects the system displays the internal reference names.

The speed of calculation mostly depends on the CPU and bus transfer speed as our algorithms require color-coded projections of the scene. For the heart model (see Fig. 7), we achieved the following frame rates at a resolution of about 800x600 pixels:

<i>CPU</i>	<i>RAM</i>	<i>GPU</i>	<i>FPS</i>
P4 2GHz	512MB	GeForce4	10
Centrino 1.6GHz	512MB	ATI 9700m	25
P4 HT 3.3GHz	1GB	ATI X800	30

## 7 Future Work

We are currently working on enhancing the quality of label layouts and improving our algorithms to extract more accurate skeletons in order to increase the readability of text strokes. Furthermore, we plan to integrate additional aesthetic metrics for evaluating their impact in a user study.

**Label Layout:** A label re-classification (internal  $\leftrightarrow$  external label) implies large incoherencies in the label layout. Moreover, connecting lines of external labels may cross internal labels. We also aim at implementing multi-line labels for both label placement modes.

**Skeletonization:** A huge variety of these algorithms were developed within the field of image processing (2D skeletons) and computer graphics (3D skeletons). As a definition of an optimal skeleton still lacks, one has to define application-specific quality measures. We focused on a low computational effort in order to achieve interactive frame rates, a sufficient quality and a high robustness.<sup>3</sup>

The modular architecture of our system enables us to experiment with several skeletonization algorithms. Our current implementation employs a scan-line algorithm [19] on the horizontal axis. This choice is motivated by the preferred left-to-right reading direction for on-line documents and the low computational effort and the robustness of

<sup>3</sup> Skeletonization algorithms are normally very sensitive to noise, i.e., the resulting skeleton differs heavily for even small changes in the object silhouette.

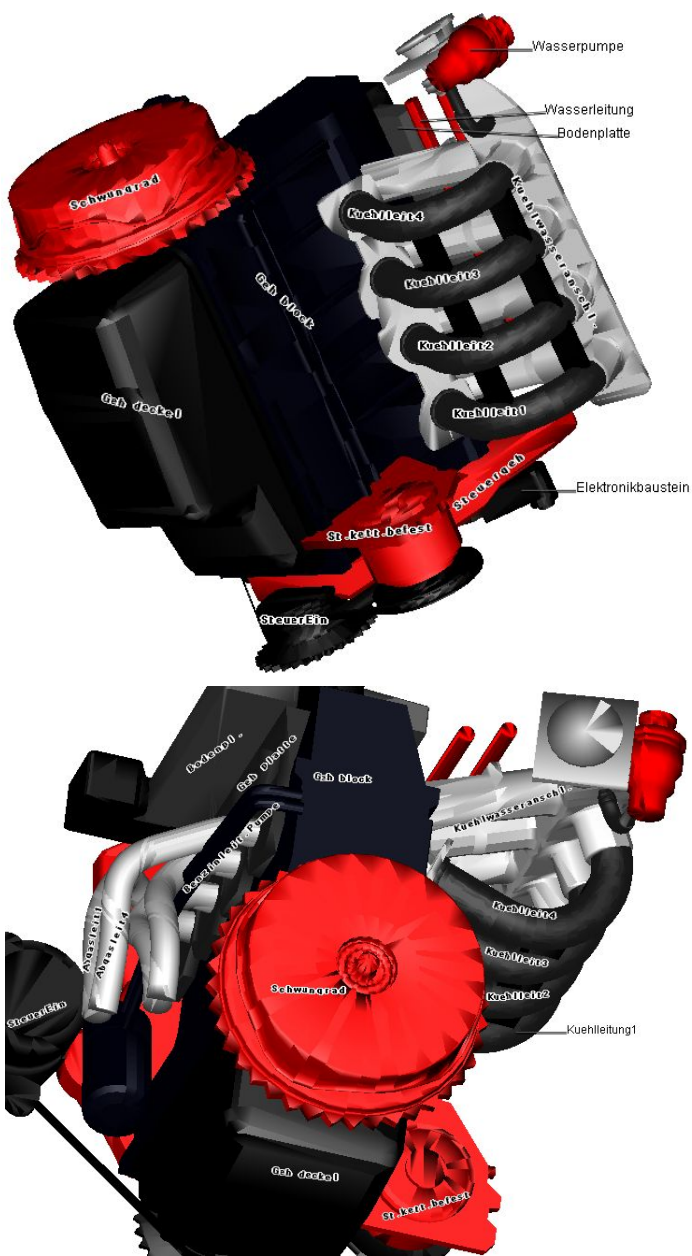
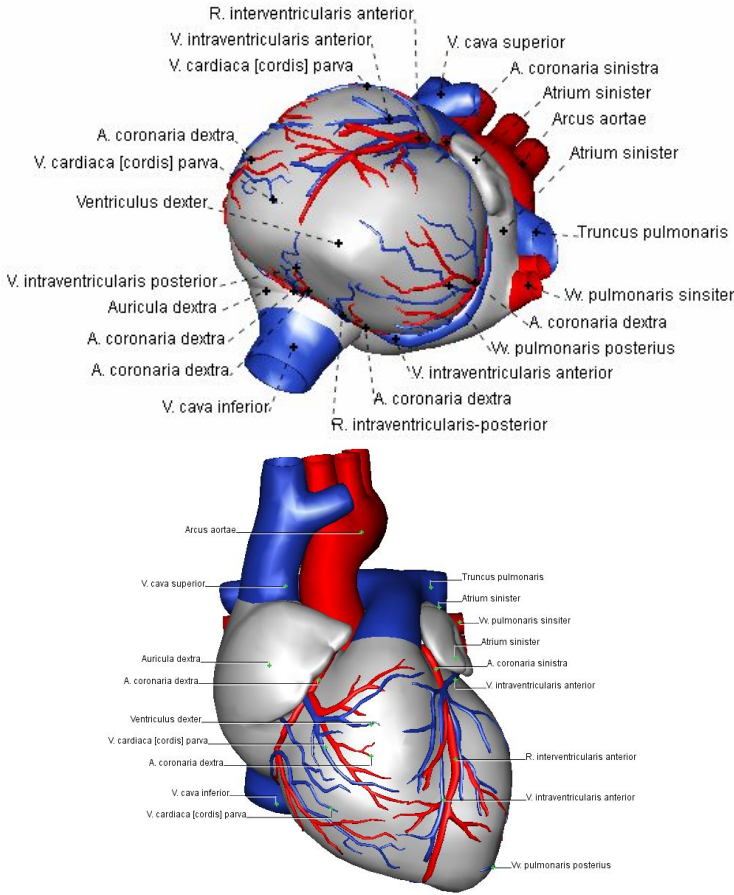


Fig. 6. Integrated label layout



**Fig. 7.** External label layout styles

this algorithm with respect to noise. The resulting discontinuities and jumps are easily removed with median filters. The more elegant distance transformation [25], as implemented in the original label layout algorithm for external labels [3], was replaced by the scan-line algorithm to increase the frame-rate.

**Evaluation:** We are preparing several user studies with our label layout prototype to evaluate different settings (like the weights and the coherent movement of the letters) and styles. Moreover, we want to study how interactive labeling can applications support different learning tasks. This would also incorporate dynamic content within labels as introduced by PREIM’s ZOOM ILLUSTRATOR [21].

Since our algorithms internally work on color-coded 2D images, we can compare the layout of human experts and automatically generated layouts. Therefore, we plan an evaluation where users should score the different layout styles, specify criteria of their pros and cons in several applications and identify hand-made layouts from auto-



matically generated layouts. Further evaluations aim at comparing the impact of layout features on the efficiency in different search tasks (reading time, error rate).

## 8 Conclusion

The impact of a beautiful illustration might be spoiled due to a poor label layout. An aesthetic illustration raises the interest of viewers, whereas an appealing label layout smoothly guides the viewer's attention. Based on the principle "form follows function", the design of an aesthetic label layout is an efficient way to increase the functionality (speeding up search tasks) of technical and scientific illustration. We propose several metrics of aesthetic attributes which are used to improve the visual balance of an initial label layout through an efficient real-time optimization process. While users can specify a preferred layout style for internal and external labels, the classification and the layout algorithm balance all constraints in order to achieve a readable, unambiguous, aesthetic, and coherent layout.

The main focus and the new contribution of our approach is to increase the coherency of the label layout during user interactions. The label layout styles presented in this paper support an interactive exploration of complex spatial configurations within a 3D browser by efficient real-time layout algorithms and a minimal layout flickering. As functional requirements and aesthetic attributes often interfere and are also subject to individual preferences, our implementation was also designed to evaluate their impact on the layout.

## References

1. M. Agrawala, D. Phan, J. Heiser, J. Haymaker, J. Klingner, P. Hanrahan, and B. Tversky. Designing Effective Step-By-Step Assembly Instructions. In *Proc. of the 30th Int. Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH 2003)*, pages 828–837, 2003.
2. M. Agrawala and C. Stolte. Rendering Effective Route Maps: Improving Usability Through Generalization. In *Proc. of the 28th Annual Conf. on Computer Graphics and Interactive Techniques (SIGGRAPH 2001)*, pages 241–250, 2001.
3. K. Ali, K. Hartmann, and T. Strothotte. Label Layout for Interactive 3D Illustrations. *Journal of the WSCG*, 13:1–8, 2005.
4. R. Azuma and C. Furmanskim. Evaluating Label Placement for Augmented Reality View Management. In *Proc. of the IEEE and ACM Int. Symposium on Mixed and Augmented Reality (ISMAR 2003)*, pages 66–75, 2003.
5. M. Bekos, M. Kaufmann, A. Symvonis, and A. Wolff. Boundary Labeling: Models and Efficient Algorithms for Rectangular Maps. In *Proc. 12th Int. Symposium on Graph Drawing (GD'04)*, pages 49–59, 2004.
6. B. Bell, S. Feiner, and T. Hllerer. View Management for Virtual and Augmented Reality. In *Proc. of Symposium on User Interface Software and Technology (UIST'01)*, pages 101–110, 2001.
7. J. Christensen, J. Marks, and S. Shieber. An Empirical Study of Algorithms for Point-Feature Label Placement. *ACM Transactions on Graphics*, 14(3):203–232, 1995.
8. J. Diepstraten, D. Weiskopf, and T. Ertl. Interactive Cutaway Illustrations. *Computer Graphics Forum*, 22(3):523–532, Sept. 2003.

9. D. Drschlag, I. Petzold, and L. Plmer. Automatic Labelling of Areas in Thematic Maps. In *Proc. of the 21st Int. Cartographic Conf. (ICC'03)*, 2003.
10. S. Edmondson, J. Christensen, J. Marks, and S. Shieber. A General Cartographic Labeling Algorithm. *Cartographica*, 33(4):13–23, 1997.
11. J.-D. Fekete and C. Plaisant. Excentric Labeling: Dynamic Neighborhood Labeling for Data Visualization. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, pages 512–519, 1999.
12. T. Goetzmann, K. Ali, K. Hartmann, and T. Strothotte. Interactive Labels. 2005. (submitted).
13. H. Gray. *Anatomy of the Human Body*. Lea & Febiger, Philadelphia, 20th edition, 1918.
14. E. Imhof. Positioning Names on Maps. *The American Cartographer*, 2(2):128–144, 1975.
15. W. Li, M. Agrawala, and D. Salesin. Interactive Image-Based Exploded View Diagrams. In *Proc. of Graphics Interface 2004 (GI'04)*, pages 203–212, 2004.
16. J. Marks and S. Shieber. The Computational Complexity of Cartographic Label Placement. Technical Report TR-05-91, Center for Research in Computing Technology, Harvard University, 1991.
17. R. Mayer. *Multimedia Learning*. Cambridge University Press, 2003.
18. I. Petzold, G. Grger, and L. Plmer. Fast Screen Map Labeling — Data Structures and Algorithms. In *Proc. of the 21st Int. Cartographic Conf. (ICC'03)*, 2003.
19. D. Peuquet. An Examination of Techniques for Remote Formatting Digital Cartographic Data. Part 1: The Raster to Vector Process. *Cartographica*, 18(1):34–38, 1981.
20. I. Pitt, B. Preim, and S. Schlechtweg. An Evaluation of Interaction Techniques for the Exploration of 3D-Illustrations. In *Software-Ergonomie'99. Design von Informationswelten*, pages 275–286, 1999.
21. B. Preim, A. Raab, and T. Strothotte. Coherent Zooming of Illustrations with 3D-Graphics and Text. In W. Davis, M. Mantei, and V. Klassen, editors, *Proc. of Graphics Interface '97*, pages 105–113, 1997.
22. F. Ritter, B. Berendt, B. Fischer, R. Richter, and B. Preim. Virtual 3D Jigsaw Puzzles: Studying the Effect of Exploring Spatial Relations with Implicit Guidance. In *Mensch & Computer 2002*, pages 363–372, 2002.
23. F. Ritter, H. Sonnet, K. Hartmann, and T. Strothotte. Illustrative Shadows: Integrating 3D and 2D Information Display. In *Proc. of 2003 Int. Conf. on Intelligent User Interfaces (IUI'03)*, pages 166–173, 2003.
24. A. Rogers. *Textbook of Anatomy*. Churchill Livingstone, Edinburgh, 1992.
25. F. Shih and Y. Wu. Fast Euclidean Distance Transformation in Two Scans Using a 3x3 Neighborhood. *Computer Vision and Image Understanding*, 93(2):195–205, 2004.
26. J. Sobotta, R. Putz, and R. Pabst, editors. *Sobotta: Atlas of Human Anatomy. Volume 2: Thorax, Abdomen, Pelvis, Lower Limb*. Williams & Wilkins, Baltimore, 12. English edition, 1997.

# A Smart Algorithm for Column Chart Labeling

Sebastian Müller<sup>1</sup> and Arno Schödl<sup>2</sup>

<sup>1</sup> Institut für Informatik, Humboldt-Universität, 10099 Berlin, Germany

`sebastian@muellerdigital.net`

<sup>2</sup> think-cell Software, 10115 Berlin, Germany

`aschoedl@think-cell.com`

**Abstract.** This paper presents a smart algorithm for labeling column charts and their derivatives. To efficiently solve the problem, we separate it into two sub-problems. We first present a geometric algorithm to solve the problem of finding a good labeling for the labels of a single column, given that some other columns have already been labeled. We then present a strategy for finding a good order in which columns should be labeled, which repeatedly uses the first algorithm for some limited look-ahead. The presented algorithm is being used in a commercial product to label charts, and has shown in practice to produce satisfactory results.

## 1 Introduction

Column charts in all their variations are among the most common forms of data visualization. The need for an automated solution arises when charts are frequently updated and manually placed labels have to be repeatedly rearranged. So far, standard commercial software does not offer automatic and intelligent chart labeling.

In the research community, different areas of automatic layout problems have been considered [1]. Cartographic labeling problems of point, line and area features have traditionally received the most attention. Most variants of these labeling problems are NP-hard [2] [3] [4]. In particular, for point features, various approaches have been tried, among them gradient descent [5], rule-based systems [6] [7], simulated annealing [8] and treating it as a combinatorial optimization problem [9] [10]. Typically, the optimization criterion is either the number of labels, which can be placed without collisions, or the maximum font size for which all labels can still be placed. Alternatively, if labels are allowed to be positioned away from their features and connected by a line, minimizing the length of connectors is a good goal function [11]. A set of constraints forbids label-label and label-point intersections. More recently, several rule-based algorithms for the point feature labeling problem have been developed which prune impossible or redundant solutions from the solution space and then search the remaining solutions with greater efficiency [12] [13]. There also exist approximative algorithms guaranteed to run in polynomial time [14] [15].

Unfortunately, in practice, applying general point feature labeling to column chart labeling gives unsatisfactory results, and no specialized algorithms have

been published. The number of labels and their size is usually set by the user, and must be respected by the algorithm. To be aesthetically pleasing, the solution must look orderly, which rules out the typical label clouds generated by point feature algorithms. Finally, the solution needs to be computed at interactive speed, for example to be integrated into a commercial presentation software like PowerPoint.

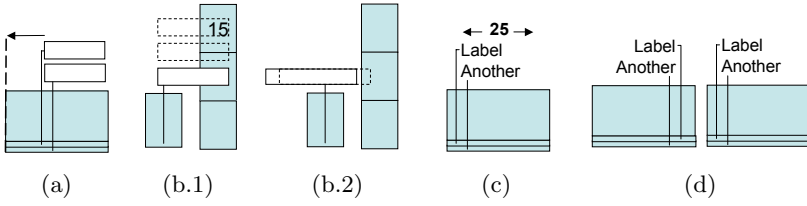
For each segment to be labeled, a few labeling positions must be considered. If there is enough space, the label should be put into the column segment. When the label is only slightly too large to fit into the segment, putting the label into a little box of background or segment color can increase legibility. If the label collides with labels of segments above or below, labels can be horizontally staggered. To avoid further collisions, some labels can be put to the side of the column if the space between columns is wide enough. Finally, for very wide labels or in case of small column spacing, labels can be arranged above or below their columns.

Although our implementation considers all possible positions described above, this paper focuses on the final, and most difficult placement of stacking labels above or below their columns. For an orderly appearance, we arrange the labels belonging to one column in a stack, where labels are in the same order as their corresponding segments. Each stack can have its connectors on the left or right side, which poses a combinatorial problem (Fig. 1 (d)). When adding a placement quality metric, the problem turns into an optimization problem. The solution is constrained by disallowing label-label and label-segment intersections.

## 2 Problem Definition

As the name implies, a column chart is made of a number of columns which are composed of multiple segments. Each segment has a label and some of these labels must be placed as a block on top of the column. In addition, each column can have a sum label which must be placed directly on top of the block of segment labels but can be moved independently in the horizontal direction. The problem is to find placements for all labels on top of their columns, and decide for each block of labels if it should be right- or left-aligned, with the goal of minimizing the total height of the chart with its labels. The following constraints which are illustrated in Fig. 1 must be observed:

- a) On the aligned side, the block labels cannot be moved over the column edge to leave room for connector lines to be drawn.
- b) The individual labels must not intersect other labels and can only intrude other columns as long as they do not intersect horizontal segment boundaries. Non-intruding solutions are preferred over intruding ones if they are otherwise of equal quality. Allowing segment intersections here may seem odd, but we found it to significantly improve appearance.
- c) The sum label is always placed on top of the column and other labels.



**Fig. 1.** The different types of choices and constraints: (a) Labels cannot be moved over the column edge on the aligned side; (b.1) Labels can intersect neighboring columns but not segment bounds or labels; (b.2) if possible, a solution which does not intersect a neighboring column is preferred; (c) the sum label is always placed on top and can be moved independently in the horizontal direction; (d) Labels can be right- or left-aligned.

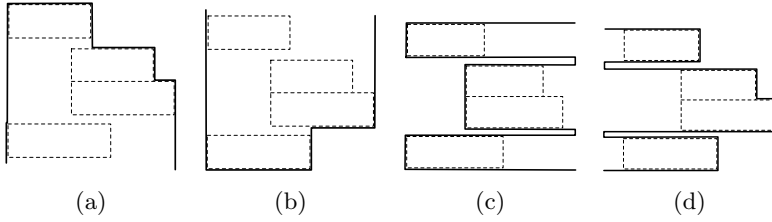
### 3 Finding a Local Solution

To make the labeling problem tractable, we separate it into two sub-problems. The first is finding the best placement of a single block of labels belonging to one column, given that some blocks of labels of other columns have already been placed. Given such an algorithm, the second problem is a strategy in which order columns should be processed. We start by describing an algorithm for the first, more geometric problem.

In order to find the best placement of a block of labels, collisions with other, already placed label blocks and the chart segments themselves must be avoided. More specifically, we must compute the best 2D position, represented by a shift vector  $\mathbf{V}$  relative to the optimal position of the label block right above the column. This vector is computed by procedure `CALCULATEBESTPOSITION` given the label block, the set of all labels and, implicitly, the chart segments. As a quality criterion for `CALCULATEBESTPOSITION` we are using the distance of the label block from its desired position right above the column it belongs to.

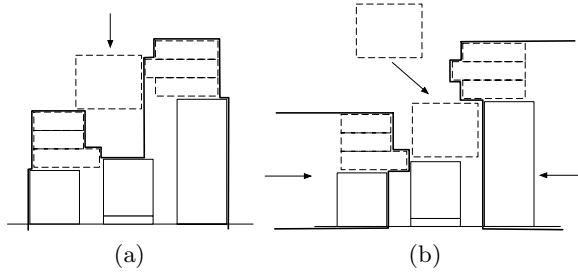
A *frontier* is a structure which provides an efficient way of finding this optimal position. It is essentially a function defined over a set of geometrical shapes  $S$ . This function can be defined as  $f(x) = \max\{y \mid (x, y) \in s \wedge s \in S\}$ . That means a frontier only contains the maxima in y-direction of all shapes contained in  $S$ . The function  $f(x)$  and the shapes in  $S$  are represented as piece-wise linear approximations. The frontier provides the two operations `ADD` and `DISTANCE` which allow adding a shape to  $S$  and computing the distance between a given shape and the frontier, respectively. Of course, we can similarly define frontiers for the other three directions (Fig. 2).

A trivial way to compute the position of a label block would be to create a frontier containing the outline of the chart itself and of all already placed labels and to let the block of labels fall down at a certain x-coordinate. However, using this strategy places the labels on top of the first obstacle they encounter, even if there is sufficient space below this obstacle to fit the label. This space cannot be modelled by a vertically oriented frontier. However, we can use horizontally



**Fig. 2.** Possible frontiers: (a) vertical orientation, growing to the left, (b) vertical orientation, growing to the right, (c) horizontal orientation growing down (d) horizontal orientation growing up.

oriented frontiers to look for a label position between the boundaries of neighboring columns and other labels: a left one named  $F_l$  growing towards higher x-coordinates and  $F_r$ , the right frontier growing towards the lower x-coordinates. Both approaches are compared in Fig. 3. Then, we have to devise an efficient way to find a space between those bounds which is wide enough to fit the labels and which is closest to the desired label position immediately above the column.



**Fig. 3.** Calculating the label block position: (a) letting the labels fall down vertically; (b) letting the labels slide into their position between horizontal frontiers resulting in a much better solution.

The function `MOVEOVERFRONTIER` shown on page 131 moves a list of shapes  $C$  over frontier  $F$  and computes a function  $g(x)$  which for every  $x$  returns the maximum  $y$  so that moving  $C$  by  $(x, y)$  will make  $C$  touch but not intersect the frontier  $F$ . For the left frontier  $F_l$ , which can be defined as

$$F_l(y) = \max\{x \mid (x, y) \in S\}, \quad (1)$$

the function `MOVEOVERFRONTIER` would compute the function

$$g(y) = \max\{x \mid \forall (x', y') \in C \wedge F_l(y' + y) \geq x' + x\}. \quad (2)$$

As all shapes in  $C$ , frontier  $F$  and function  $g(y)$  are represented as piecewise linear approximations, we can compare every shape in  $C$  and the frontier  $F$  line segment by line segment. Every line of every shape is moved over every

line segment of  $F$ . Depending on the frontier segment, two cases have to be distinguished (l. 5 and l. 10). In both cases we calculate two vectors, one which moves the line along the frontier segment and another which moves the line over the end of the frontier segments. We can now regard these vectors as simple line segments and add them to our new frontier  $\overline{F}$ . In a regular frontier  $F$ , for every position  $y$ ,  $F$  describes the maximum x-coordinate of all contained shapes. In the newly formed frontier  $\overline{F}$ , for every  $y$ ,  $\overline{F}$  describes the  $x$ -coordinate which makes the shapes in  $C$  touch but not intersect  $F$ .

---

**Algorithm 1.** MoveOverFrontier Algorithm

---

**Input:** A list of shapes  $C$  and a frontier  $F$   
**Output:** A frontier  $\overline{F}$  containing the vectors which move all shapes in  $C$  along  $F$

```

1 frontier  $\overline{F}$ ;
2 foreach shape  $\in C$  do
3   foreach line  $\in$  shape do
4     foreach lineFrontier  $\in F$  do
5       if lineFrontier.from.x  $\leq$  lineFrontier.to.x then
6         ptFrom = line.to - lineFrontier.to;
7         ptTo = line.to - lineFrontier.from;
8          $\overline{F}$ .Add( Line(ptFrom, ptTo));
9          $\overline{F}$ .Add( Line(ptTo, ptFrom - (line.to - line.from), ptFrom));
10      else
11        ptFrom = line.from - lineFrontier.from;
12        ptTo = line.to - lineFrontier.from;
13         $\overline{F}$ .Add( Line(ptFrom, ptTo));
14         $\overline{F}$ .Add( Line(
15          ptFrom - ( lineFrontier.to - lineFrontier.from ), ptFrom));
16      end
17    end
18  end
19 end
20 return  $\overline{F}$ ;

```

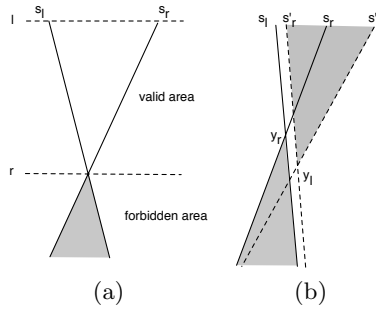
---

Using the frontier and the MOVEOVERFRONTIER algorithm we can now implement the procedure CALCULATEBESTPOSITION as follows. We calculate for every label block which has to be placed the two frontiers  $F_l$  and  $F_r$  representing the rightmost and leftmost bounds of the chart's parts to the left or the right of the labels' column.

Then we create the frontiers  $\overline{F}_l$  and  $\overline{F}_r$  by moving our label block  $L$  over  $F_l$  and  $F_r$ . The frontiers  $\overline{F}_l$  and  $\overline{F}_r$  define the space of possible solutions for the label block which is available between  $F_l$  and  $F_r$ . For every move by a given  $y$ , moving the label block by the resulting  $\overline{F}_l(y)$  or  $\overline{F}_r(y)$  will make it touch the frontiers  $F_l$  or  $F_r$ , respectively. If for a given  $y$ ,  $\overline{F}_r(y) > \overline{F}_l(y)$ , then there is not enough space between  $F_l$  and  $F_r$  at position  $y$  to fit the label.

Because the sum label is allowed to move in horizontal direction independently of the block of segment labels, we repeat the same procedure for the sum label, thereby creating two more frontiers  $\overline{F}_l'$  and  $\overline{F}_r'$  from  $F_l$  and  $F_r$ .

We then iterate over the four frontiers  $\overline{F}_l, \overline{F}_r, \overline{F}_l', \overline{F}_r'$  at once. The 4-tuple of line segments  $(s_l, s_r, s_l', s_r') \in \overline{F}_l \times \overline{F}_r \times \overline{F}_l' \times \overline{F}_r'$  defines a part of our solution space. We subdivide segments as necessary so that all segments  $(s_l, s_r, s_l', s_r')$  have the same start and end  $y$ -coordinates. In this area we search for a shift vector  $\mathbf{V}$  which is closest to our preferred initial position, ie. closest to a shift  $(0, 0)$ , and a vector specifying the sum label position  $\mathbf{V}'$ .  $\mathbf{V}$  and  $\mathbf{V}'$  are constrained to share the same  $y$ -coordinate, but may have different  $x$ -coordinates, reflecting independent horizontal movement of the sum label.



**Fig. 4.** (a) The two frontier segments  $s_l$  and  $s_r$  are shown as an example.  $y_l$  and  $y_r$  define the limits of the solution space. (b) Considering all four frontier segments the both pairs can intersect in a way, that our solution space is empty and  $y_l > y_r$  holds.

We find the solution  $\mathbf{V}$  for the label block in the space between the segments  $s_l$  and  $s_r$ . The sum label solution  $\mathbf{V}'$  is in the space defined by  $s_l'$  and  $s_r'$ . Intersections between the segments (cf. Fig. 4) indicate that there is no room at this position to fit the labels. Let  $[l, r]$  be the interval which limits the space of valid solutions between  $s_l$  and  $s_r$  and let  $[l', r']$  be the interval which limits the space of valid solutions between  $s_l'$  and  $s_r'$ . We calculate the left and right boundary of solution space  $y_l = \max(l, l')$  and  $y_r = \min(r, r')$ . If  $y_l > y_r$ , our solution space is empty because we have two disjoint solution spaces for the label block and the sum label. If the segments did not intersect, it is still possible that the solution space is empty because the segments overlap in the whole interval. If  $y_l < y_r$ , we shorten all four segments  $s_l, s_r, s_l', s_r'$  to the interval  $[y_l, y_r]$ .

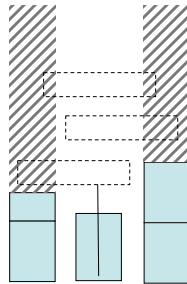
Now, the solution  $\mathbf{V}$  is the point closest to  $P = (0, 0)$  and point  $P$  can either be inside the polygon defined by  $s_l$  and  $s_r$  or the solution is the point on the polygon outline closest to  $P$  which is easily computed by projecting the point on all four line segments of the rectangle's outline and by choosing the point closest to  $P$  of the four solutions obtained. The sum label solution  $\mathbf{V}'$  with the same  $x$ -coordinate as  $\mathbf{V}$  is then guaranteed to exist because the solution space is not empty and it is easily found between  $s_l'$  and  $s_r'$ .



All of the above is actually done twice, for the label block aligned on the left and on the right. We then choose the alignment with a position closer to  $(0, 0)$ .

## 4 Determining the Labeling Order

After having explained how a single label block can be placed, the second, more strategic problem of determining a labeling order remains. A simple approach is to iterate simultaneously from the left and right over the set of all columns, labeling the left labels right-aligned and the right labels left-aligned. At each step of the iteration we place the left or right label block, whichever has the better placement. This approach guarantees that all label blocks can actually be placed without collisions: When proceeding on the left side, we use right-aligned label blocks, which have their connecting lines on the right, and which can only intersect labels to their left, which have already been placed. Collisions with these labels can be avoided by placing the new label block high enough. The same holds for the right side. The advantage of this simple approach is that it always yields a solution, the disadvantage, however, is that the solution will often have the form of a pyramid with labels stacked on top of each other towards the center. To improve the algorithm, we can order the columns by their height and label them starting with the lowest column. Unfortunately, placing label blocks in an arbitrary order can prevent a column to be labeled at all, if all room above the column is taken up by other labels. We avoid this dead-end by inserting an artificial shape above each unlabeled column, blocking all space above the columns, as illustrated in Fig. 5. Although the average results of this variant are much better, there is still an easily identifiable worst-case example. If the columns increase in height from the left to the right, the labels will also be stacked one on top of the other.



**Fig. 5.** The look-ahead of the MULTIFRONTIERLOOKAHEAD algorithm: the left and right column which have not yet been labeled are blocked in order to guarantee that it can still be labeled in the future. The middle column cannot be labeled in this step because the label is too wide.

To avoid this problem, instead of predetermining the order of labeling, at each step, we calculate the best label block position for each column, given the already placed labels. We again avoid the dead-end described above by blocking the space above unlabeled columns. After calculating the best possible positions for each column, we choose the column with the lowest top label block border to be the one to place its block at its calculated position. The rationale behind this criterion is to free as much room above columns as possible as early as possible, to give more space to future label placements.

---

**Algorithm 2.** MULTIFRONTIERLOOKAHEAD Algorithm
 

---

```

1   $L \leftarrow$  list of label blocks;
2  foreach  $l \in L$  do
3     $l.top \leftarrow$  highest y-value of the label block's outline;
4     $l.labeled \leftarrow$  false;
5     $l.hasvalidsolution \leftarrow$  false;
6  end
7  while  $\exists l \in L : l.labeled = \text{false}$  do
8     $l_{opt} \leftarrow nil$ ;
9     $V_{opt} \leftarrow nil$ ;
10   foreach  $l \in E$  do
11     if  $l.labeled = \text{false}$  then
12       if  $l.hasvalidsolution = \text{false}$  then
13          $V = \text{CalculateBestPosition}(l, L)$ ;
14         if  $V$  is valid then
15            $l.hasvalidsolution \leftarrow \text{true}$ ;
16         end
17       end
18       if  $l.hasvalidsolution = \text{true}$ 
19          $\wedge (l_{opt} = nil \vee l.top + V.y < l_{opt}.top + V_{opt}.y)$  then
20            $l_{opt} \leftarrow l$ ;
21            $V_{opt} \leftarrow V$ ;
22         end
23     end
24   end
25   PlaceLabel ( $l_{opt}$ ,  $V_{opt}$ );
26    $l.hasvalidsolution \leftarrow$  false for all labels  $l$  which intersect with  $l_{opt}$ 
27 end

```

---

We found that in many cases, this heuristic ordering is actually close to the order that a human would use to place labels. The algorithm is guaranteed to find a solution, which in the worst case deteriorates to the simple pyramid of stacked labels described in the beginning of this section.

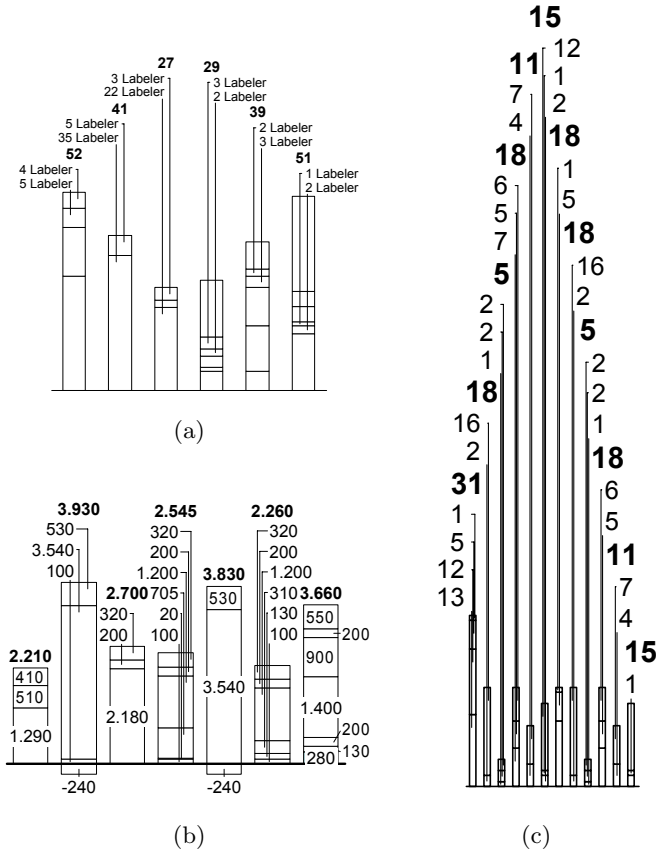
## 5 Extensions

One constraint has been ignored in the solution so far. Figure 1 shows labels which overlap their neighboring columns, which is allowed, as long as they do not overlap contained labels or segment boundaries. The geometric algorithm makes the solution easy and it has been omitted to facilitate the presentation. When in procedure `CALCULATEBESTPOSITION` the Frontiers  $F_l$  and  $F_r$  are created, we can add the horizontal segment and column bounds and the contained labels to the Frontier, and not the segments themselves which thus may be intruded. We can compute both solutions, with and without intruding, and pick the non-intruding one if it is otherwise no worse than the intruding one. Likewise, other aesthetic constraints can be easily included into our formulation. For example, if labels should have a margin, we can inflate the shapes added to the Frontier by a small amount.

To obtain interactive speed, we use two efficiency optimizations. Line 25 of the `MULTIFRONTIERLOOKAHEAD` algorithm already shows that after placing a label  $l_{opt}$ , only the labels affected by placing  $l_{opt}$  must be recalculated. Notice that they may not only be affected by collisions with the newly placed label, but also by the freeing of the blocked space above the column belonging to  $l_{opt}$ . Secondly, we can exploit the fact that the Frontiers  $F_l$  and  $F_r$  in `CALCULATEBESTPOSITION` can be calculated recursively: We can compute  $F_l^n$  for column  $n$  by copying  $F_l^{n-1}$  from column  $n - 1$  and adding the shapes of the next column. After placing a new label block, only the affected range of Frontiers must be recalculated.

## 6 Evaluation

The worst case for our greedy algorithm is a column chart in form of an inverted pyramid where the columns are getting successively lower towards the middle (Fig. 6 (a)). If all labels are too wide to fit over their column the algorithm will start labeling them from the left- and rightmost column effectively creating a pyramid of stacked labels mirroring the pyramid of columns. Under the constraints specified in the problem definition this is the best labeling. A human user would possibly try to find a solution which violates as few constraints as possible. However, this case can typically be resolved by making the chart a little bit wider. This example is also not typical for a column chart because all labels have the same width and all labels are wider than their respective columns. Figure 6 (b) shows a more typical representative of column charts which is labeled optimally. The positive and negative columns are treated as separate problems and the negative values are labeled downwards. In the second column the value 3.540 is stacked on top whereas in the fourth column it is not. In the second column the value 100 has to be moved to the top because it is too large and intersects a big portion of the segment below. As a result, the 3.540 is moved to the top too. Otherwise, there would not be enough space to fit the connecting line in the segment without intersecting the label 3.540. Case (c) is an extreme example which shows, that the algorithm always finds a solution even when the chart becomes very small.



**Fig. 6.** Labeling examples with timing information. Timing tests were made on an Athlon 64 3800+ machine: (a) Worst-case example where all labels are wider than their respective columns (Labeling took 60.5 ms); (b) Typical case with an optimal solution (115 ms); (c) Extreme case with a very small chart (157 ms).

## 7 Conclusion

The presented algorithm has been implemented as part of the commercial charting software think-cell chart, and customers, comparing it to manually labeled charts, are satisfied with its performance. Even for worst-case examples, the algorithm provides solutions which pass as good enough in practice. In general, we found that for practical applications worst-case performance is more important than the average performance metrics often used in academic papers.

## Acknowledgements

This work has been funded by think-cell Software GmbH. We want to thank Prof. Dr. Hans-Dieter Burkhard and Dr. Gabriele Lindemann from the Dept. of Artificial Intelligence at Humboldt-University, Berlin.

## References

- [1] Steven Lok and Steven Feiner. A survey of automated layout techniques for information presentations. *Proceedings of SmartGraphics, Hawethorne, USA*, 2001.
- [2] J. Marks and S. Shieber. The computational complexity of cartographic label placement. Advanced Research in Computing Technology TR-05-91, Harvard University, 1991.
- [3] M. Formann and F. Wagner. A packing problem with applications to lettering of maps. *Proceedings of the 7th Annual ACM Symposium on Computational Geometry*, pages 281 – 288, 1991.
- [4] C. Iturriaga and A. Lubiw. Np-hardness of some map labeling problems. Technical Report CS-97-18, University of Waterloo, 1997.
- [5] S. Hirsch. An algorithm for automatic name placement around point data. *The American Cartographer*, 9(1):5 – 17, 1982.
- [6] J. S. Doerschler and H. Freeman. A rule-based system for dense-map name placement. *Communications of the ACM*, 34(1):68 – 79, 1992.
- [7] A. C. Cook and C. B. Jones. A prolog rule-based system for cartographic name placement. *Computer Graphics Forum*, 9(2):109 – 126, 1990.
- [8] J. Christensen, J. Marks, and S. Shieber. An empirical study of algorithms for point-feature label placement. *ACM Transactions on Graphics*, 14(3):203 – 232, 1995.
- [9] R.G. Cromley. An lp relaxation procedure for annotating point features using interactive graphics. *Proceedings of Auto-Carto 7*, pages 127 – 132, 1985.
- [10] S. Zoraster. The solution of large 0-1 integer programming problems encountered in automated cartography. *Operations Research*, 38(5):752 – 759, 1990.
- [11] Michael A. Bekos, Michael Kaufmann, Antonios Symvonis, and Alexander Wolff. Boundary labeling: Models and efficient algorithms for rectangular maps. In János Pach, editor, *Proceedings of 12th Int. Symposium on Graph Drawing*, volume 3383 of Lecture Notes in Computer Science, pages 49 – 59. Springer Verlag, 2005.
- [12] A. Wolff. *Automated Label Placement in Theory and Practice*. PhD thesis, Freie Universität Berlin, 1999.
- [13] F. Wagner, A. Wolff, V. Kapoor, and T. Strijk. Three rules suffice for good label placement. *Algorithmica Special Issue on GIS*, 2000.
- [14] Pankaj K. Agarwal, Marc van Kreveld, and Subash Suri. Label placement by maximum independent set in rectangles. *Proceedings of the 9th Canadian Conference on Computational Geometry*, pages 233 – 238, 1997.
- [15] M. van Kreveld, T. Strijk, and A. Wolff. Point labeling with sliding labels. *Computational Geometry: Theory and Applications*, 13:21 – 47, 1999.

# Usability Comparison of Mouse-Based Interaction Techniques for Predictable 3d Rotation

Ragnar Bade<sup>1</sup>, Felix Ritter<sup>2</sup>, and Bernhard Preim<sup>1</sup>

<sup>1</sup> University of Magdeburg, Universitätsplatz 2, D-39106 Magdeburg, Germany  
bade@isg.cs.uni-magdeburg.de

<sup>2</sup> MeVis - Center for Medical Diagnostic Systems and Visualization,  
Universitätsallee 29, D-28359 Bremen, Germany

**Abstract.** Due to the progress in computer graphics hardware high resolution 3d models may be explored at interactive frame rates, and facilities to explore them are a part of modern radiological workstations and therapy planning systems. Despite their advantages, 3d visualizations are only employed by a minority of potential users and even these employ 3d visualizations for a few selected tasks only. We hypothesize that this results from a lack of intuitive interaction techniques for 3d rotation. In this paper, we compare existing techniques with respect to design principles derived by clinical applications and present results of an empirical study. These results are relevant beyond clinical applications and strongly suggest that the presented design principles are crucial for comfortable and predictable interaction techniques for 3d rotation.

## 1 Introduction

3d visualization offers a great potential for medical applications, such as therapy planning and surgical education. 3d models of relevant anatomic structures may be derived by means of CT or MRI data. Due to the progress in computer graphics hardware high resolution 3d models may be explored at interactive frame rates, and facilities to explore them are a part of modern radiological workstations. Despite their advantages, 3d visualizations are only employed by a minority of potential users and even these employ 3d visualizations for a few selected tasks only. Instead, medical doctors prefer to "read" CT or MRI data slice by slice.

We hypothesize that one reason for this situation is the lack of intuitive interaction techniques for 3d rotation. Medical doctors attempt to explore geometric models systematically (looking at each part of an organ for example). Hence, they require a predictable behavior which enables them to anticipate the effect of an interaction and to return to a previous viewing direction.

Software systems for geometric modelling, Computer-Aided Design and surgery planning provide a variety of interaction techniques for 3d rotation<sup>1</sup>. In this

---

<sup>1</sup> For readability we will use "rotation techniques" as a synonym for "interaction techniques for 3d rotation".

paper, we compare such techniques with respect to design principles inspired by clinical applications. The comparison of 3d rotation techniques as well as the discussed principles are relevant beyond clinical applications. We focus on users controlling 3d rotation by means of a standard 2d mouse and do not consider dedicated 3d input devices since these are still rare.

We evaluate direct manipulation techniques because we assume that this interaction style is superior to indirect controllers such as scroll wheels for 3d rotation. Bimanual (two-handed) interaction techniques [2] as well as interaction techniques which require 3d pointing devices are not considered.

First, we present a list of principles for predictable and convenient rotation techniques in Sect. 2. Subsequently, we discuss state of the art direct manipulation rotation techniques for 2d pointing devices in Sect. 3 and in Sect. 4, we report and discuss results of an empirical user study.

## 2 Design Principles for 3d Rotation Techniques

Based on our experience in clinical applications (cp. [13]), we identified the following four general principles as crucial for predictable and pleasing rotation techniques:

1. *Similar actions should provoke similar reactions.*
2. *Direction of rotation should match the direction of 2d pointing device movement.*
3. *3d rotation should be transitive.*
4. *The control-to-display (C/D) ratio should be customizable.*

The first is a general principle for human-computer interaction which emphasizes the importance of a predictable and reproducible behavior. As an example, the same mouse movement should not result in varying cursor movements.

The second principle applies to stimulus-response (S-R) compatibility and kinesthetic correspondence [4] between the direction of user action (e.g. mouse movement) and the direction of computer reaction (e.g. object rotation).

The third principle completes the first two by taking into account the natural transitive behavior of movements in an Euclidean space (human world). Since a pointing device movement from point A to point B and then to point C ends at the same location as a direct movement from A to C, this should also be true for the corresponding reaction of such an action. We identified this principle as crucial to enable users to return to the initial viewing position.

While the first three principles are important for a predictable behavior, the fourth principle takes user and application needs into account. Customizing the control-to-display (C/D) ratio [5] is necessary to find the best compromise between speed and accuracy according to the task and user preferences and is therefore crucial for speed, accuracy and user satisfaction.

### 3 State of the Art 3d Rotation Techniques

In this section, we analyze and compare different state of the art rotation techniques. The list of principles in Sect. 2 is utilized to formally classify these techniques and to point out existing drawbacks. Since this comparison focuses on usability concerns, details of the underlying mathematics and implementation are not covered but can be retrieved from the cited publications.

#### 3.1 Chen et al.’s Virtual Trackball

The virtual trackball (VT) implemented by Chen et al. [6], the so-called Virtual Sphere, can be described as a 3d sphere located behind the viewport. The 2d viewport location of the moving mouse is projected onto this sphere to get corresponding 3d points on the sphere’s surface. Thus, the 2d motion of the mouse is mapped to a 3d rotation from one projected point on the sphere’s surface to a second one. (See Henriksen et al. [8] for a mathematical description of this and all other discussed techniques.) Unfortunately, the resulting rotation axis is not necessarily perpendicular to the mouse displacement vector [8]. The VT implemented by Bell [3] as well as Shoemake’s VT [18] (both discussed further on) corrects this. Therefore, we will not further investigate Chen et al.’s VT.

#### 3.2 Bell’s Virtual Trackball

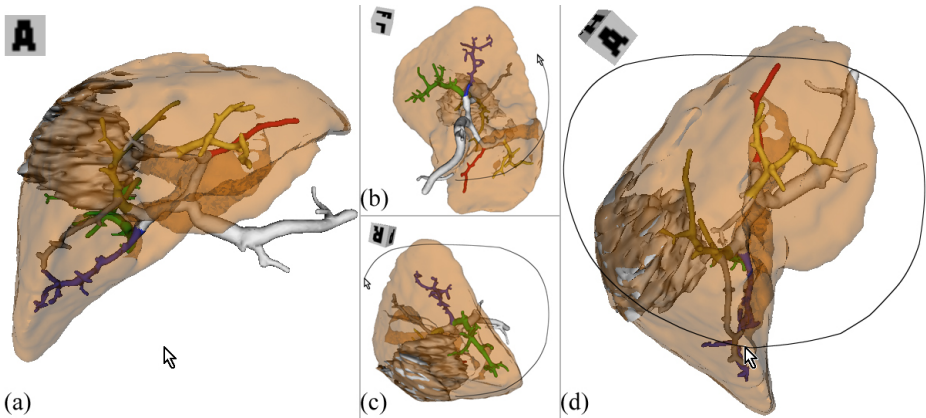
The VT implemented by Bell [3] is an improved version of Chen et al.’s VT [6] [8]. Instead of projecting the 2d mouse location onto a sphere, Bell projects it to a combination of a sphere and a hyperbola in 3d space. Thus, rotating a 3d scene using this approach appears very smooth. Nevertheless, with respect to our principles list Bell’s VT fails in several points.

Moving the mouse by a vector  $(\Delta x, \Delta y)$  at different locations on the viewport results in different rotations. As one example, a horizontal mouse movement in the center of the viewport rotates the 3d scene about the up-vector of the virtual camera. In contrast, the same movement at the bottom or top border of the viewport rotates the 3d scene about the viewing-axis of the virtual camera. Thus, even simple horizontal or vertical mouse movements may result in unpredictable rotations – Bell’s VT violates Principle 1.

Even if similar actions do not provoke similar reactions and the rotation direction changes over the viewport space using this VT, the direction of the 3d rotation remains similar to the direction of the 2d mouse movement. For example, moving the mouse at the top border of the viewport to the right rotates the 3d scene clockwise and moving the mouse at the bottom border of the viewport to the right rotates the 3d scene counterclockwise. This behavior violates Principle 1 but fulfills Principle 2.

In contrast to Principle 3, a combination of two rotations using Bell’s VT is not transitive. Moreover, the 3d scene starts to tumble if the mouse is moved on a circular trajectory. Thus, closed loops of mouse motion may not produce the expected closed loops of rotation (see Fig. 1). However, even in this case the





**Fig. 1.** Bell's VT is used to rotate a liver with tumor and vessels. (a)-(d) show different time steps. As illustrated, closed loops of mouse motion (black line) may not produce the expected closed loops of rotation (cp. (a), (d)).

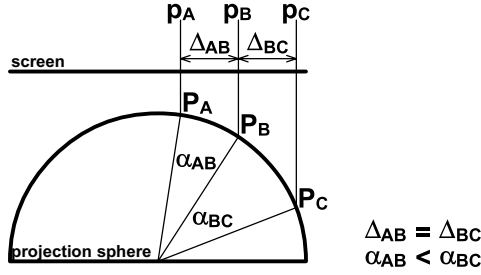
rotation direction remains similar to the direction of mouse movement. As an example, while moving the mouse on a counter-clockwise circular trajectory, the object spins counter-clockwise too.

Due to the underlying mathematics of Bell's VT, it is not possible to customize the C/D ratio (Principle 4) in a convenient manner. Since using this technique the C/D ratio depends on the viewport size – changing the size on the screen (e.g. changing window size of the 3d view) results in a modified C/D ratio. As one result, the smaller the viewport the faster the 3d scene is rotated and vice versa. In addition, the rotation using Bell's VT is limited to  $90^\circ$  from the center of the viewport to its borders. As described before, this fixed rotation ratio results in a dependency between viewport size and C/D ratio. This limitation can be eliminated by introducing a scale factor for the calculated rotation angles. Unfortunately, this results in very different rotation behaviors as will be discussed in Sect. 3.3.

Despite these problems, Bell's VT is integrated in a wide range of applications especially due to its implementation in the Open Inventor graphics toolkit [17], [19].

### 3.3 Shoemake's Virtual Trackball: ArcBall

Shoemake's VT [18], the so-called Arcball, is a special version of Chen et al.'s VT (see [8]). Shoemake as well utilizes the projection of the mouse location onto a sphere to calculate rotation axis and angle. However, due to a carefully chosen scale factor for the calculated rotation angles using Shoemake's VT rotations are transitive. Thus, Shoemake's VT fulfills Principle 2 and 3 but similar to Bell's VT the Arcball violates Principle 1. However, in contrast to Bell's VT using Shoemake's Arcball the 3d scene is rotated twice about the viewing-axis if the mouse is moved once around the projection sphere, which may lead to confusion and slightly violates Principle 2.



**Fig. 2.** Discontinuity of the resulting rotation angle of Shoemake’s VT at the rim of the projected sphere: The distance  $\Delta_{AB}$  between viewport points  $p_A$  and  $p_B$  equals the distance  $\Delta_{BC}$  between  $p_B$  and  $p_C$ . However, the resulting rotation angle  $\alpha_{AB}$  between projected points  $P_A$  and  $P_B$  is smaller than the angle  $\alpha_{BC}$  between  $P_B$  and  $P_C$ .

Due to the underlying mathematics (similar to Bell’s VT), Shoemake’s VT does not provide a convenient way to customize the C/D ratio. The smaller the viewport the faster the 3d scene is rotated. Changing the size of the underlying projection sphere inversely proportional to the viewport size would fix this problem. However, this would either result in large viewport regions characterized by a rotation restricted to the viewing-axis of the camera or in a projection sphere much larger than the viewport hampering the 3d rotation about all axes. Similar to Bell’s VT the rotation by means of Shoemake’s VT is limited to  $180^\circ$  from the center of the viewport to its borders. Thus, the C/D ratio depends on the viewport size. As suggested for Bell’s VT, this limitation can be eliminated by introducing a different scale factor for the calculated rotation angles. However, using a factor that halves the rotation speed of Shoemake’s VT will cause its behavior to adapt to Chen et al.’s and Bell’s VT (see [8]) which violates Principle 3. Furthermore, due to the projection geometry of Shoemake’s VT (a sphere) the C/D ratio changes from the center of the viewport to the rim of the projected sphere as illustrated in Fig. 2. As one result, the rotation seems to snap to the rim with a loss of user control and accuracy. This phenomenon has also been reported in [8] as a discontinuity of the rotation.

Even though the implementation of Shoemake’s VT (see [10]) is simpler compared to Bell’s VT, it is not wide-spread.

### 3.4 Two-Axis Valuator Trackball

In the Two-Axis Valuator approach (cp. [6]) the horizontal mouse movement is mapped to a rotation about the up-vector of the virtual camera and the vertical mouse movement is mapped to a rotation about the vector perpendicular to the up-vector and the view-vector of the camera. Diagonal mouse movement is mapped to a combined rotation about both axes.<sup>2</sup> As one result, no explicit rota-

<sup>2</sup> In a right-handed coordinate system with the negative z-axis being the view-vector the horizontal rotation is about the y-axis and the vertical rotation is about the x-axis of the camera.

tion about the view-vector is possible and thus, it is often separately controllable by pressing another mouse button or an additional control key.

In contrast to all other discussed techniques, here the same mouse movement always results in the same rotation. Thus, the Two-Axis Valuator is the first (and only) approach that fulfills Principle 1. Furthermore, it fulfills Principle 2 since the direction of rotation always equals the direction of the mouse movement. In contrast to Principle 3, a combination of two rotations using the Two-Axis Valuator is not transitive. Furthermore, a clockwise circular mouse movement rotates the 3d scene counter-clockwise – Principle 2 is violated in this special case. Due to the direct mapping of the 2d mouse movement to a corresponding rotation about the two axes, the C/D ratio can be easily customized by changing the mapping factor. Thus, the Two-Axis Valuator fulfills Principle 4.

The Two-Axis Valuator is widely used in applications such as the Visualization Toolkit (VTK) [12]. This is probably due to the simple implementation. Another reason might be the fact that the Two-Axis Valuator fulfills three of the principles stated in Sect. 2 (with a slight deduction concerning Principle 2).

### 3.5 Two-Axis Valuator with Fixed Up-Vector

For 3d modeling purposes (e.g. in 3d-Studio-Max [7]) or associated tasks (e.g. in Deep Exploration [16]) often a special version of the Two-Axis Valuator is applied. This version uses the world's up-vector for horizontal rotation. Since this vector is fixed, the approach is called Two-Axis Valuator with fixed up-vector.

Using a Two-Axis Valuator with fixed up-vector offers the advantage of transitive rotations – Principle 3 is fulfilled. Unfortunately, this leads to discrepancies between mouse movement direction and rotation direction – Thus, Principle 1 is violated. Furthermore, the rotation direction is contrary to the direction of mouse movement if the 3d scene is rotated upside down – Principle 2 is violated too. Nevertheless, Principle 4 is fulfilled due to the direct mouse movement mapping of the Two-Axis Valuator.

As mentioned before, the Two-Axis Valuator with fixed up-vector is used in 3d modeling applications. This is probably due to the transitivity of rotations and the customizable C/D ratio which are both crucial for the level of precision in those applications.

### 3.6 Tabular Comparison

As discussed in Sect. 3.1 – 3.5 each rotation technique exhibits different behavior and different features such as transitivity or a direct stimulus-response (S-R) compatibility and kinesthetic correspondence. None of the rotation techniques fulfills all principles from Sect. 2 (see Tab. 1).

**Table 1.** Comparison of state of the art rotation techniques

Rotation Technique:	Bell's VT	Shoemaker's VT	Two-Axis Valuator	Two-Axis Valuator with fixed up-vector
Principle 1	–	–	+	–
Principle 2	+	+	+/-	–
Principle 3	–	+	–	+
Principle 4	–	–	+	+

## 4 Evaluation of 3d Rotation Techniques

In Section 3 we systematically compared 3d rotation techniques with respect to a list of design principles which are inspired by clinical applications (see Sect. 2). As one result, we identified certain drawbacks of the discussed techniques and pointed out different usability crux of each of them. However, their influence on usability aspects such as performance, accuracy and user satisfaction is widely unknown.

As Henriksen et al. [8] stated, only four studies have empirically evaluated virtual trackballs ([6], [9], [15], [11]). Three of them (Chen et al. [6], Hinckley et al. [9] and Partala [15]) used relatively simple rotation tasks. In these “orientation matching tasks” the users were instructed to rotate an object to a target orientation presented by an image of this object in the desired orientation. Only Jacob and Oliver [11] stepped beyond and included a more complex inspection task (e.g. find the number of windows in a house). Since they found differences between rotation techniques even for different simple tasks, rotation techniques have to be empirically evaluated for more complex tasks to reveal trade-offs among them. Moreover, we notice that all these studies used only common, well-known objects with an inherent orientation (e.g. a house or a human head) which is strongly different from real world tasks for example in therapy planning and surgical education applications. Here, the shape or appearance of an object (e.g. a tumor) may not provide conclusive hints of the object’s orientation.

In the following we present our empirical usability study of the discussed rotation techniques from Sect. 3 which focusses on the intuitiveness and appropriateness of these techniques for complex rotation tasks with complex-shaped objects.

### 4.1 Evaluation Goals

We attempt to compare different state of the art rotation techniques with focus on the degree of intuitive usage, performance and user satisfaction in conjunction with complex scan and hit tasks. This allows us to identify suitable rotation techniques for further usage, deeper evaluations and as a basis for future improvements.

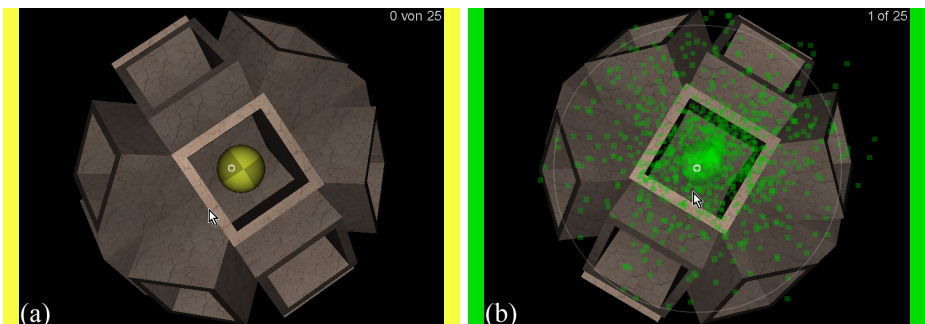
## 4.2 Evaluation Strategy

Concerning therapy planning and surgical education applications, rotating a 3d scene is only a tool for solving various tasks. In this working environment it is crucial that the rotation techniques are comfortable and predictable. Therefore, our evaluation strategy focuses on the intuitive usage. No instructions, explanations and demonstrations of the usage and functions of the rotation techniques were presented to our subjects. The evaluation program could be downloaded from our project web-site [1] which provides further information and instructions on how to use this program as well.

**Apparatus.** The evaluation was carried out by each user on his/her own computer system. Thus, users where not confronted with unfamiliar input devices and foreign device performance settings. For the evaluation we used a world-in-hand metaphor which let users feel to rotate the 3d scene and not the camera which had proven to be best suited for orbiting (cp. [15]). The evaluation program including questionnaire and all rotation techniques is implemented in Macromedia Director 8.5 [14].

**Task.** In order to compare the rotation techniques, we identified the systematic exploration (scanning for a target) of a mostly unknown 3d scene in conjunction with rotating the target to the center of the view (e.g. for further inspections) as crucial. Thus, subjects were asked to perform a series of scan and hit (search and shoot) tasks.

Subjects were confronted with a complex-shaped symmetric and unknown object without an inherent orientation and had to scan this object for a target by rotating it (see Fig. 3). Once the target had been located they had to rotate it to the center of the screen and to shoot the target by pressing the space bar or the enter key (as they prefer). They were instructed that speed is important.



**Fig. 3.** Screenshots of the experiment software. Subjects rotated the 3d scene (e.g. by Bell's VT (a) and Shoemake's VT (b)) to locate the targets and to position them in the center of the viewport to shoot them. The viewport center is marked by a small white circle.

A defined level of accuracy was forced by the system such that the targets had to be rotated to a position inside a small circle on the screen. The allowed fault tolerance between the target center and the center of shooting had been  $13^\circ$ . The number of bullets required to shoot all targets was counted to quantify inaccuracy in solving the task.

After shooting a bullet the user received both visual and aural feedback indicating a hit (explosion of the target) or miss (impact on the rotated object, target remains in position to retry).

**Design.** The evaluation was designed in the form of a game with different levels which had to be completed in random order. In each level, one of the rotation techniques from Sect. 3 is used to scan an object for 25 targets which had to be localized and shoot. Only one target at a time is randomly positioned in one of 18 funnels of the object that had to be scanned. After one target had been shoot another one appeared with a short delay.

**Procedure.** At the beginning all subjects had to answer a questionnaire concerning their experience and familiarity with a mouse as input device, with 3d tools, 3d games and 3d rotation. Afterwards, subjects were given a general description of the experimental procedure.

Before entering a new level, a training scene was presented to become familiar with the rotation technique of the current level and to adjust the rotation speed to personal needs (by dragging a small slider) if the rotation technique provides this. At the beginning of each training session, subjects were instructed to click and drag with the mouse to rotate the scene and to press the space bar or enter key to shoot. The practice was neither limited in time nor in number of targets and could be terminated by the subjects whenever they felt familiar enough with the rotation technique. Subsequently, in the level subjects had to shoot 25 targets as fast as possible. To ensure reliable completion times, no adjustments of the rotation speed were allowed within a level.

After each level, subjects had to assess how comfortable they felt with hitting the targets and how close the rotation technique in this level followed their expectations. They were also informed about their time required to finish the level and a top ten of the fastest subjects in this level was presented. The same procedure was then repeated for the remaining levels (rotation techniques). The entire procedure took about 30 minutes.

**Subjects.** Forty-two unpaid subjects (30 male, 12 female) took part in the evaluation. All of them had long-term experience with computer mice. On a scale from 1(low) to 7(high) all but four assessed their experience as 7 whereas these four subjects assessed their experience as 5 or 6, respectively. In contrast, all subjects had very different 3d navigation and rotation experience. To avoid a bias caused by different input device types only users which employed a mouse were included in the study.

### 4.3 Statistical Analysis

We first applied the One-Sample Kolmogorov-Smirnov-Test to test if the given data distributions are significantly different from a normal distribution. Task completion times exhibit normal distribution ( $p \geq .05$ ) whereas the number of shots required, experience and rotation technique assessment data differ significantly from a normal distribution. All subjects had to complete one level after another thus, we used paired samples tests to analyze differences between the rotation techniques.

To analyze differences in task completion time, we used the paired samples t-test. To analyze differences in the number of shots required, we applied the Wilcoxon-Signed-Ranks-Test and to analyze differences in the ordinal scaled assessment of the rotation techniques, the Sign-Test was employed.

### 4.4 Evaluation Results and Discussion

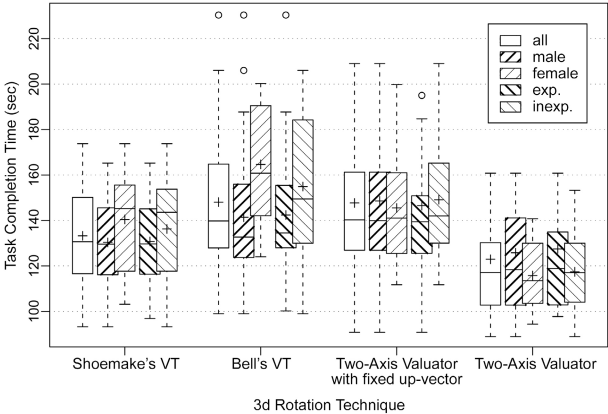
Statistics on the mean task completion times are shown in Tab. 2 and in Fig. 4, respectively. Comparison of completion times (Tab. 3) revealed that with high significance ( $p \leq .001$ ) users performed faster with Shoemake's VT and the Two-Axis Valuator than with the Two-Axis Valuator with fixed up-vector and Bell's VT. Furthermore, the Two-Axis Valuator was significantly ( $p \leq .05$ ) faster than Shoemake's VT. Between Bell's VT and the Two-Axis Valuator with fixed up-vector no significant difference could be observed.

Using Shoemake's VT and the Two-Axis Valuator subjects required significantly ( $p \leq .05$ ) more shots to complete the task compared to Bell's VT (Tab. 3, Fig. 5). However, users performed significantly faster. We did not observe any other significant correlations of shots required between the rotation techniques.

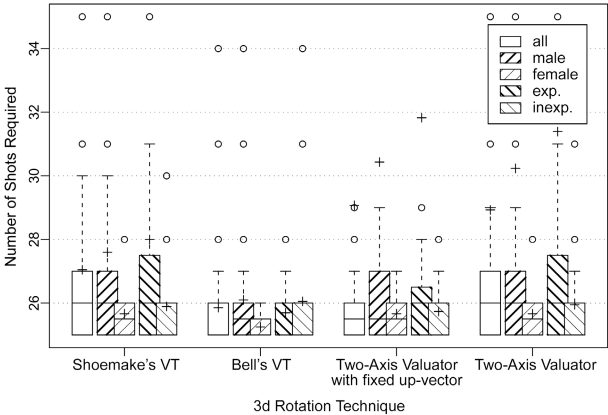
Statistics on the assessment of all interaction techniques concerning (1) comfortable task completion and (2) whether the behavior was perceived as predictable are shown in Fig. 6. As expected, the Two-Axis Valuator that fulfills most of the principles presented in Sect. 2 was assessed significantly ( $p \leq .01$ ) best with respect to both questions. In contrast, Bell's VT, Shoemake's VT and

**Table 2.** Mean completion time for each rotation technique with separate means achieved by male (30), female (12), experienced (23) and inexperienced (19) subjects

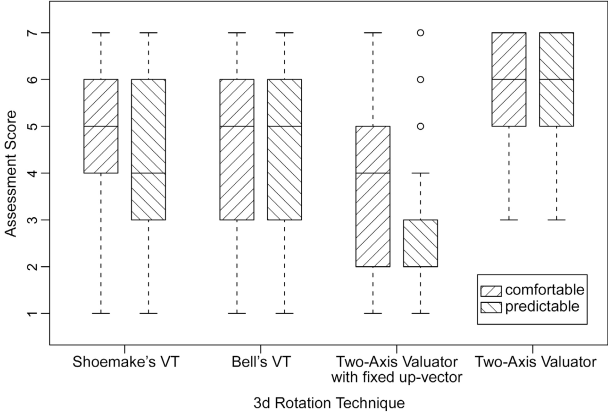
Rotation Technique	Mean Completion Time (in sec)				
	all subjects	males	females	experienced	inexperienced
Bell's VT	148.06	141.42	164.67	142.39	154.92
Shoemake's VT	133.24	130.37	140.42	130.75	136.26
Two-Axis Valuator	122.92	125.79	115.75	127.48	117.39
Two-Axis Valuator with fixed up-vector	147.71	148.59	145.50	146.54	149.12



**Fig. 4.** Boxplot of completion times for all rotation techniques for all, male (30), female (12), experienced (23) and inexperienced (19) subjects



**Fig. 5.** Boxplot of number of shots required for all rotation techniques for all, male (30), female (12), experienced (23) and inexperienced (19) subjects



**Fig. 6.** Boxplot of assessment score for all rotation techniques



**Table 3.** Comparison of rotation techniques regarding completion time and shots required

Comparison	T-Test		Wilcoxon-Test	
	Completion Time		Shots Required	
Bell's VT vs. Shoemake's VT	t = -4.240	p = .000	Z = -2.083	p = .037
Bell's VT vs. Two-Axis Valuator	t = 5.360	p = .000	Z = -2.181	p = .029
Bell's VT vs. Two-Axis Valuator with fixed up-vector	t = 0.080	p = .937	Z = -0.796	p = .426
Shoemake's VT vs. Two-Axis Valuator	t = 2.231	p = .031	Z = -0.585	p = .558
Shoemake's VT vs. Two-Axis Valuator with fixed up-vector	t = -3.053	p = .004	Z = -0.766	p = .444
Two-Axis Valuator vs. Two-Axis Valuator with fixed up-vector	t = 7.153	p = .000	Z = -1.187	p = .235

the Two-Axis Valuator with fixed up-vector were assessed very similar concerning question (1). Concerning question (2), Bell's VT and Shoemake's VT were assessed similar but significantly ( $p \leq .001$ ) superior to the Two-Axis Valuator with fixed up-vector.

A separate analysis for males and females revealed a slightly better performance for male subjects for any of the 3d rotation techniques. Using Bell's VT, females were significantly ( $p \leq .05$ ) slower. The survey also revealed that all females were less experienced in 3d navigation, 3d rotation, 3d modeling programs and 3d games. To verify if that explains the detected differences, we subdivided the subject pool into two groups according to their experience. A separate analysis of these two groups revealed no significant performance differences. Consequently, the significant difference in completion time using Bell's VT between males and females suggests that sex significantly influences user performance with Bell's VT.

To summarize, the Two-Axis Valuator as well as Shoemake's VT performed best even though using both techniques more shots had been required to hit all targets and to complete the task. Furthermore, subjects attested the Two-Axis Valuator to be the most convenient rotation technique which strongly supports our principles list.

## 5 Conclusion

We compared Bell's VT, Shoemake's VT, the Two-Axis Valuator and the Two-Axis Valuator with fixed up-vector by means of an experiment where we measured task completion times and shots required for complex scan and hit tasks. Our experiment design is inspired by clinical applications involving the complete exploration of arbitrary positioned and oriented complex shaped objects. A questionnaire was employed to assess whether 3d rotation techniques are perceived as comfortable and predictable.

Our evaluation revealed significant performance and user satisfaction differences between common 3d rotation techniques. The Two-Axis Valuator turned out to be the best 3d rotation technique while Bell's VT which is wide-spread in general 3d user interface toolkits is rated lowest with respect to speed and satisfaction. We are currently working on a refined version of interactive 3d rotation since we hypothesize that still better user performance is possible by an appropriate combination of flexibility and user-guidance.

## References

1. Bade, R.: Evaluation of 3d rotation techniques, <http://www.isg.cs.uni-magdeburg.de/cv/projects/LST/rotation>, April (2005)
2. Balakrishnan, R. Kurtenbach G.: Exploring bimanual camera control and object manipulation in 3D graphics interfaces. *Proc. of ACM SIGCHI'99* (1999) 56–62
3. Bell, G.: Bell's Trackball. [http://members.tripod.com/professor\\_tom/index.html](http://members.tripod.com/professor_tom/index.html), (1988)
4. Britton, E.G. Lipscomb, J.S. Pique, M.E.: Making nested rotations convenient for the user. *Proc. of SIGGRAPH'78* (1978) 222–227
5. Chapanis, A. Kinkade, R.: Design of Controls. H. vanCott, R. Kinkade (ed.) *Human Engineering Guide to Equipment Design*, US Government Printing Office (1972)
6. Chen, M. Mountford, S.J. Sellen, A.: A study in interactive 3D rotation using 2D control devices. *Computer Graphics*, **22(4)** (1988) 121–129
7. Discreet: Discreet 3D-Studio-Max. <http://www.discreet.com/3dsmax>, April (2005)
8. Henriksen, K. Sparring, J. Hornbæk, K.: Virtual trackballs revisited. *IEEE Transactions on Visualization and Computer Graphics*, **10(2)** (2004) 206–216
9. Hinckley, K. Tullio, J. Pausch, R. Proffitt, D. Kassel, N.: Usability analysis of 3D rotation techniques. *Proc. of ACM UIST'97* (1997) 1–10
10. Hultquist, J.: *A Virtual Trackball*. Graphics Gems, Academic Press (1990) 462–463
11. Jacob, I. Oliver, J.: Evaluation of Techniques for Specifying 3D Rotations with 2D Input Device. *Proc. of HCI'95* (1995) 63–76
12. Kitware Inc: VTK Home Page. <http://www.vtk.org>, April (2005)
13. Krueger, A. Tietjen, C. Hintze, J. Preim, B. Hertel, I. Strauss, G.: Interactive Visualization for Neck Dissection Planning. *Proc. of IEEE EuroVis'05*, (2005) 295–302
14. Macromedia: Macromedia Director. <http://www.macromedia.com/software/director>, April (2005)
15. Partala, T.: Controlling a Single 3D Object: Viewpoint Metaphors, Speed and Subjective Satisfaction. *Proc. of INTERACT'99*, IOS Press (1999) 486–493
16. Right Hemisphere: Deep Exploration. <http://www.righthemisphere.com/products/dexp>, April (2005)
17. SGI: SGI - Developer Central Open Source | Open Inventor. <http://oss.sgi.com/projects/inventor>, April (2005)
18. Shoemake, K.: ARCBALL: A user interface for specifying three-dimensional orientation using a mouse. *Proc. of Graphics Interface* (1992) 151–156
19. Strauss, P.S. Carey, R.: An object-oriented 3D graphics toolkit. *Proc. of SIGGRAPH'92* (1992) 341–349

# Multi-level Interaction in Parametric Design

Robert Aish<sup>1</sup> and Robert Woodbury<sup>2</sup>

<sup>1</sup> Bentley Systems, Incorporated  
robert.aish@bentley.com

<sup>2</sup> Simon Fraser University, School of Interactive Arts and Technology, 14th Floor,  
Central City Tower, 13450 102 Avenue, Surrey, BC, Canada, V3T 5X3  
rw@sfu.ca  
<http://www.siat.sfu.ca>

**Abstract.** Parametric design systems model a design as a constrained collection of schemata. Designers work in such systems at two levels: definition of schemata and constraints; and search within a schema collection for meaningful instances. Propagation-based systems yield efficient algorithms that are complete within their domain, require explicit specification of a directed acyclic constraint graph and allow relatively simple debugging strategies based on antecedents and consequents. The requirement to order constraints appears to be useful in expressing specific designer intentions and in disambiguating interaction. A key feature of such systems in practice appears to be a need for multiple views onto the constraint model and simultaneous interaction across views. We describe one multiple-view structure, its development and refinement through a large group of architecture practitioners and its realization in the system Generative Components.

## 1 Architectural Practice and Parametric Design

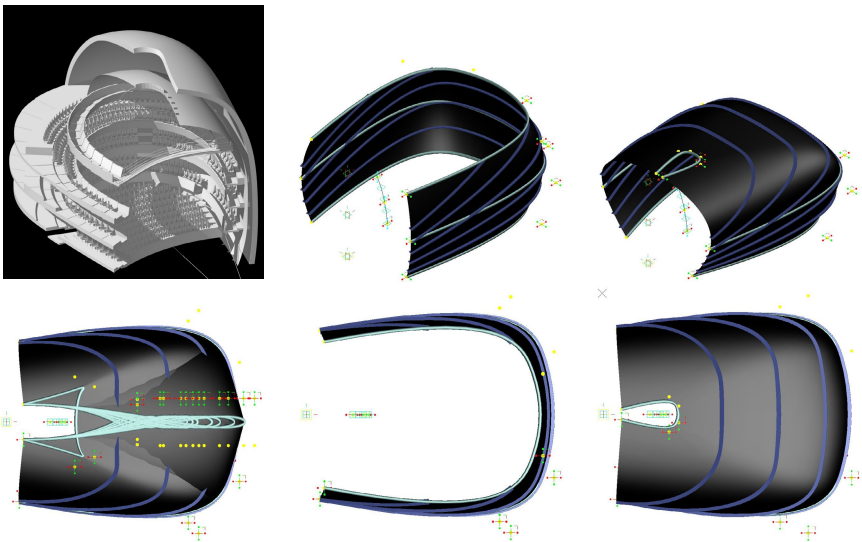
Conventional CAD systems focus design attention on the representation of the artifact being designed. Currently industry attention is on systems in which a designed artifact is represented parametrically, that is, the representation admits rapid change of design dimensions and structure. Parameterization increases complexity of both designer task and interface as designers must model not only the artifact being designed, but a conceptual structure that guides variation. Parameterization has both positive and negative task, outcome and perceptual consequences for designers. Positively, parameterization can enhance search for designs better adapted to context, can facilitate discovery of new forms and kinds of form-making, can reduce the time and effort required for change and reuse, and can yield better understandings of the conceptual structure of the artifact being designed. Negatively, parameterization may require additional effort, may increase complexity of local design decisions and increases the number of items to which attention must be paid in task completion.

Parametric modeling has become the basis for most mechanical CAD systems and now is beginning to emerge as a tool for architectural design. While there is a general appreciation of the concepts and advantages of parametric modeling,

application to projects at the scale and complexity of buildings raises important theoretical and practical issues. In a deep sense, parametric modelling is not new: building components have been adapted to context for centuries. What is new is the parallel development of fabrication technology that enables mass customization. Building components can be adapted to their context and parametric modelling can represent both context and adapted designs. In a design market partly driven by novelty, the resulting ability to envision and construct new architectural forms rewards firms having such expertise. There are relatively few such firms, most of which have had long experience and have built substantial reputations on distinctive form and construction. But many firms and students (future practitioners) are interested. The confluence of technology and interest appears as exploration in a new design space: architecture and its supporting technologies of parametric design and fabrication are experiencing both co-development and rapid change.

Emblematic of this change is the SmartGeometry group. It comprises senior practitioners, a CAD system developer and academics. Its website leads with the declaration:

Architecture is fundamentally about relationships. Many of those relationships are geometric in nature or find a geometric expression. The SmartGeometry group has been created in the belief that Computer Aided Design lends itself to capturing the geometric relationships that form the foundation of architecture....The group is dedicated to educating the construction professions in the new skills which will be required

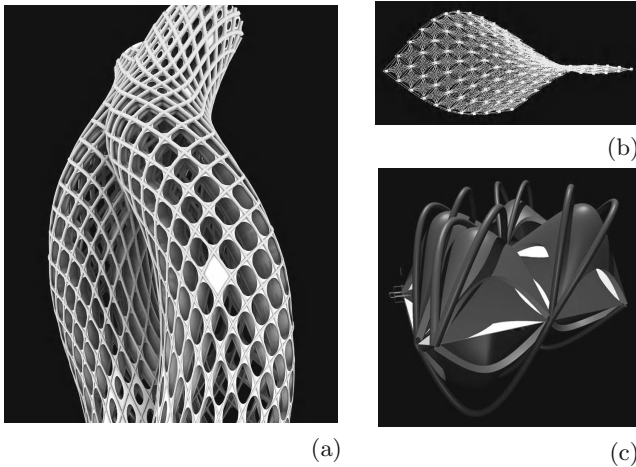


**Fig. 1.** Lars Hesselgren is a Senior Associate Partner and Director of R&D at KPF Architects. The images represent form explorations for a project currently (as of Spring 2005) under development at KPF. *(by permission of the author)*

to use these new systems effectively....The group conducts a series of schools and seminars where this new technology is explored in the context of highly experienced professionals. (*Minor verb tense changes made to the quotation.*) ([www.smartgeometry.org](http://www.smartgeometry.org))

To date, several such workshops have been conducted and workshop participants have continued to develop both designs and ideas for needed system support. The outcomes of the workshops and work that follows from them comprise designs, some intended for actual construction, some as explorations of design possibility. Figures 1 and 2 show work done by two workshop participants.

SmartGeometry works, as it must, through using (and developing) systems that support design relationships. Much of its work parallels and is influencing the development over the last eight years of a specific parametric design system Generative Components (GC) by Bentley Systems, Inc. The primary designer and developer of GC is Robert Aish. At the time of writing, GC was not on the commercial market and its development path lay outside of the Bentley product process. Its design and development have been extensively affected by members of the SmartGeometry group, the professional members of which have acted largely in addition to their professional roles inside their respective firms. Academics and graduate students have been involved in design, review and trial workshops to an



**Fig. 2.** Neri Oxman is a recent graduate of the Architectural Association in London and currently (as of Spring 2005) is at KPF Research in London. (a) Design prototype for a helical high-rise structure first modeled in a conventional 3D software package and post-rationalized in GC as a set of helical strands and tiling elements set in a cylindrical coordinate 3D space. The work represented was awarded a FEIDAD 2004 Design Merit Award. (b) & (c) Explorations in GC towards new graph nodes and an idiom of use employing a topologically-defined 'hosting environment' that dictates the position and geometrical articulation of any 4 point defined component to populate the mat. (c) A four-point based component developed to populate its hosting environment. (*by permission of the author*)

extent unusual for a corporate project. The existence of a motivated independent user community and a relatively open and relatively well-resourced system development process provides opportunities for early and frequent verification of design choices against actual need and for research connecting task to system at a scale largely not possible in research labs using systems built from scratch.

GC is a propagation-based system – implying that part of a user’s task is determining *how* particular relationships are processed in addition to specifying the relationships themselves. There are practical reasons for this. First propagation-based systems are efficient and sound. They are predictable, in that choices of what controls and what is controlling are required and are often important to designers. They are easily extensible at the local (node) level and provide multiple points of entry for more sophisticated extension. Although algorithmically simple, they are complex in use – the task of simultaneous model creation and design demands sophisticated language-level and user interfaces. The interface appears to be the principal technical obstacle to further practical use.

## 2 Propagation-Based Constraint Systems

At the representation level, a propagation-based constraint system comprises an *acyclic directed graph* and two algorithms, one for *ordering* the graph and one for *propagating values* through the graph.

The nodes of the graph are schemata, that is, they are objects containing variables and constraints amongst the variables. Variables within a node are either *independent* or *dependent* and both variables and nodes are typed. Every type of node provides an efficient *update algorithm* for updating specific values for the dependent variables subject to the constraints and given values for the independent variables, as well as *display algorithms* for displaying the node symbolically and in 3D. The arcs of the directed graph denote uses of nodes or variables from within nodes as the independent variables in a node’s constraints. An arc from node *A* to node *B* exists if a variable in *A* is identified with an independent variable in *B*. An incoming arc to a node *binds* one or more independent variables in the node. Graphs are constrained to be acyclic: any operation that introduces a cycle has undefined effect. Graphs may have arbitrarily many nodes with unbound independent variables; these are the *independent nodes* of the graph and their unbound independent variables are the *independent variables* of the graph. All other nodes and variables are *dependent*. The *antecedent* nodes of a node are those that bind its independent variables. The *consequent* nodes of a node are those that are bound by any of its variables.

A graph models a usually infinite collection of *instances*, each of which is determined by assigning values to the independent variables of the graph. Graphs are typically presented to the user through one or more of their instances.

To compute an instance, graphs are first ordered and then values are propagated through the graph. Both algorithms are theoretically simple and both are efficient. The ordering algorithm is topological ordering, which has worst case time complexity of  $O(n + e)$  where  $n$  is the number of nodes and  $e$  is the number

of edges in the graph. The propagation algorithm is also linear, with complexity  $O(n + e)$ , presuming that the internal node algorithms are  $O(1)$ .

The representation is well-known as the basis for such common tools as spreadsheets, project management tools and dataflow programming languages. In design applications, the graphs tend to be large and the nodes represent schemata of arbitrary complexity. These issues are typically addressed through *constraint languages*, which provide tools for expressing node algorithms and for building complex graphs from nodes and less complex graphs.

Propagation-based systems are the most simple type of constraint system. The first CAD system was a constraint system. Sutherland's Sketchpad [Sut63] provided both a propagation-based mechanism and a simultaneous solver based on relaxation. It was the first report of a feature that became central to many constraint languages – the *merge operator* that combines two similar structures to a single structure governed by the union of the constraints on its arguments. *Constraint management systems*, for example, Borning's Delta Blue [SMFBB93] provide primitives and constraints that are not pre-bundled together and with which the user can overconstrain the system, but is required to give some value (or utility) for the resolution of different constraints. A constraint manager does not need access to the structure of the primitives or the constraints. Rather its algorithm aims to find a particular directed acyclic graph that resolves the most highly valued constraints. *Constraint solvers* represent primitives and constraints over them with algorithms specific to the class of constraints being solved, for instance, differential equations. *Constraint logic programming* systems integrate constraint satisfaction within a logic programming framework and use constraints to limit the search process of the logic program. Different constraint domains define different classes of constraint logic programming. For instance, in  $\text{CLP}(\mathbb{R})$ , the constrained variables are real numbers and the solver is typically limited to linear relations with a delayed binding mechanism that admits solution for some non-linear problems. *Algebraic constraint* systems such as Magritte [Gos83] apply symbolic algebra either directly or to rewrite sub-graphs so that a graph suitable for propagation exists. *Constraint languages* such as ASCEND [PMW93] address the problem of building (and solving) large sets of equations in which the equation structure has significant regularity.

### 3 Parametric Modeling as a Task

A parametric model amplifies the effort of building a representation by providing an interpretation of a model as a typically infinite set of instances, each determined by a particular selection of values for the model's independent variables.

The parametric modeling task proceeds in concert with the task of creating a design. At the end of the process there exists both a graph structure and a specific instance that represents a concrete design. The main effect of separating graph and instance is to defer decisions. The graph embodies decisions about chosen relationships and defers computation of precise values (and in some cases structure) that depend on the relationships. For example, a graph representing

a roof structure may have the roof's support lines as its inputs and may produce different roofs designs depending on the location of the support lines. In contrast, non-parametric modeling systems tend to invert such a decision structure. The precise location of an object is required to model it at all and objects that depend on other objects must be explicitly modeled in precise context.

The cost of decision deferral is a higher level of abstraction in work. Users of parametric systems must explicitly develop relationships between objects and, at some level, code those into a node or graph. When a system does not support a particular relationship, it must be developed. Figure 3 shows an example of computing the shortest line between two skew lines. If such a relationship is not already part of a system it must be coded. If the needed sub-relationships are not supported, they too must be coded. Such work is necessary in practice. It turns out that the relationships that designers need to model their work comprise a set far too large and idiosyncratic to anticipate and provide as node types in a modelling environment. Designers using parametric systems must develop, and must be able to develop, relationships specific to the task at hand.

Another cost of parametric model is complexity of both representation and interface. At the representation level a designer must understand new concepts such as the graph itself, node compilation, intensionality and a set of mathematical ideas related to descriptive geometry and linear algebra. The present state of interfaces for parametric modeling systems dictates multiple, related interaction devices for different aspects of the representation. Using GC as an example, we demonstrate several novel aspects of task complexity and supporting interface.

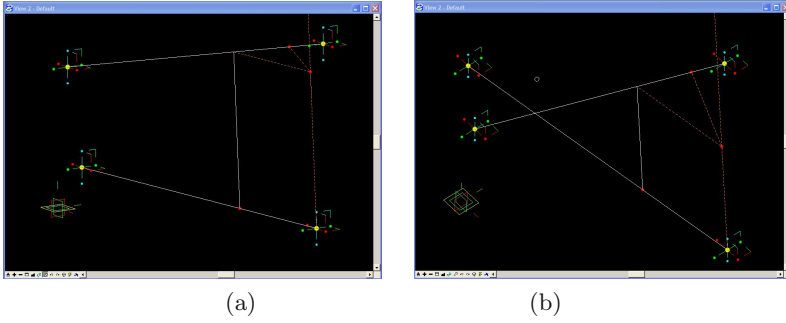
## 4 Representational Features

### 4.1 Object Compilation

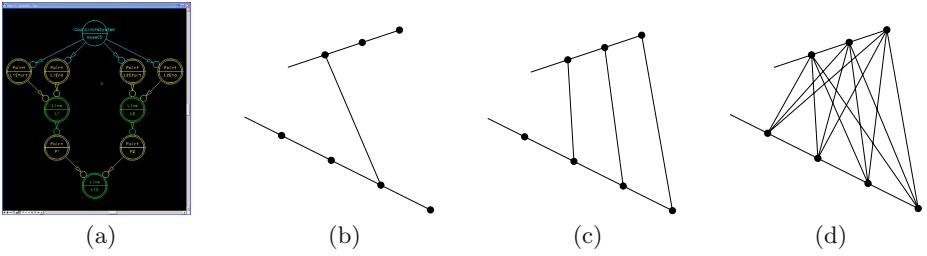
Nodes are typically defined by specifying independent and dependent variables and an update algorithm for the node. In GC the specification is a C# method where the method signature contains the independent variables, the method returns the dependent variables and the method itself specifies the update algorithm. The system uses code reflection to compile user-provided node specifications into update and display methods.

Conceptually, a node can be arbitrarily complex. To the ordering and propagation algorithms of a parametric design system a node provides independent and dependent variables and an update algorithm. A subgraph can be considered as a single node whose independent and dependent variables are those of the subgraph and whose algorithm is that of the global propagation algorithm applied to the subgraph. GC provides a second level of node compilation called *feature compilation* that, through code reflection, collapses a subgraph into a new object with its own update method. Figures 5(a) and 5(b) show the effect of compiling a graph representing a Bezier curve into a single node. Such compiled nodes can be reused in other contexts where, to a user, they represent a single modelling concept.





**Fig. 3.** Two views of a shortest line segment between skew lines. The graph (not shown) comprises a cross product, three vector projections, a converse vector projection and three construction lines.



**Fig. 4.** Single points, each expressed as a parametric point on a line. (a) A symbolic model representing a line between parametric points each on their own line. The same symbolic model represents (b), (c) and (d). A line between the two parametric points with single index values (1 & 2) for the parameters of its defining points. (c) Each of the defining points of the line has multiple parameter values. The number of lines is equal to the length of the shortest collection. (d) A line collection under the Cartesian product interpretation of a collection.

## 4.2 Objects as Collections

In GC a node's independent variables may be either *singletons* or *collections*. A collection has the interpretation that each object in the collection specifies a node in and of itself. When multiple independent variables are collection-valued, collections propagate to dependent variables in two distinct ways as shown in Figure 4. The first produces a collection of objects, of size equal to the shortest of the input collections, by using the  $i^{th}$  value of each of the input collections as independent inputs. The second produces a collection by using the Cartesian product of the input collection as arguments. In both cases, the collection finds interpretation as a single node in the graph, while its elements are accessed through an array-indexing convention. The identification of singletons and collections supports a form of programming-by-example whereby the work

done to create a single instance can be propagated to multiple instances simply by providing additional input arguments.

### 4.3 Intensionality

Intensionality means that two symbol structures can be identical in all aspects yet remain distinct. From a representational perspective a symbol is a declaration that there exists an object with the properties given in the symbol. There may exist many such objects and multiple symbols may refer to the same object. Changing the properties of a symbol does not affect the existence of the symbol – representationally it modifies the represented object, while maintaining object identity. Intensionality appears to be a necessary property in parametric modelling. In GC, it appears as *merge operator* and *variable update methods*. The merge operator is analogous to Sutherland’s [Sut63] and essentially identical to Borning’s [Bor81]. The design of update methods is modelled on Delta Blue [SMFBB93]. Update methods are uniquely distinguished by their signatures and a group of suitably designed update methods forms a clique through which a node with a method belonging to the clique may circulate, that is, users may change a node’s update method from within its clique.

## 5 Representational Views

In this section we sketch several interface views in GC. Such features have either been developed independently several times or are otherwise recurrent features of parametric design systems. It appears that multiple views and the attendant complexity in use is, at least at present, a necessary feature of parametric design systems. We include three views: 3D, symbolic graph and object (the latter being itself a composite of other views). Due to space, we omit a fourth view, namely the programmatic view through which elements may be directly programmed using the system’s API.

### 5.1 3D Interaction

Most demonstrations of parametric modelling systems are made in a *3D interactive view*. Such views comprise a 3D scene in which graphical objects are displayed. Graph nodes are represented by displaying the instance of the node associated with the current settings of the graph independent variables. The node display algorithm is determined by the node type: nodes of type *Point* display as a geometric point, nodes of type *Line* as a line. Some node types, for example *global variables*, do not have a direct representation in a 3D interactive view. Further, not all nodes need be displayed at once – it is common for nodes to have display properties such as *hidden* and *construction* that simplify a view while retaining the underlying model structure. Key node display algorithms are crafted with intent to support user intuitions about how a node is computed. For example, a point constrained to be on a curve may be displayed as a point plus a handle for dragging the point along the curve. Nodes in a 3D interactive view

may be constrained by other nodes by identifying their independent variable with either nodes their contained variables.

Users engage with a 3D interactive view to define graphs, and to manipulate independent variables to find useful graph instances that may be concretely realized. Expressing even relatively simple ideas as a graph turns out to be labour intensive and saving labour has motivated a number of interaction devices.

## 5.2 Object View: Names, Expression Entry, Functions and Scripting

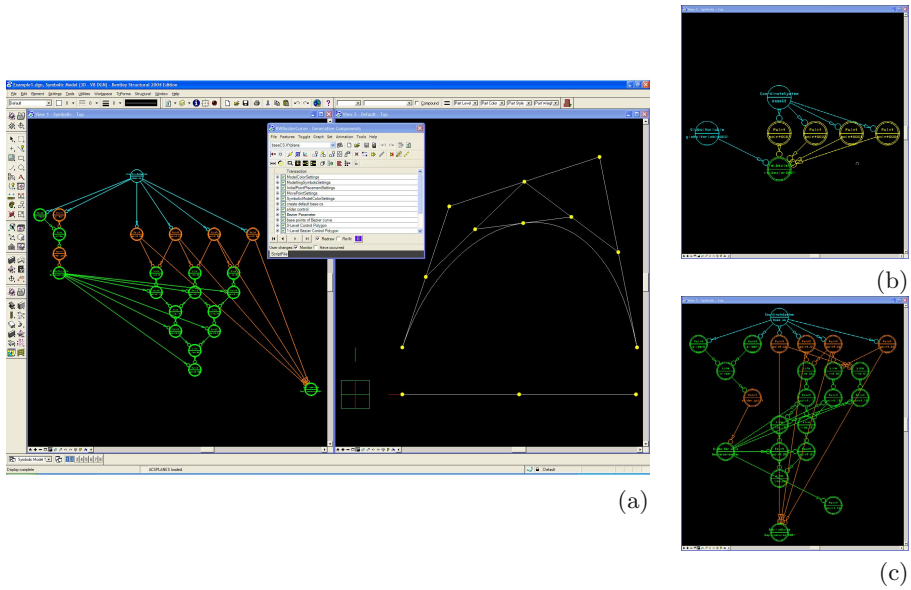
Individual variables within a node can be bound by identifying nodes or variables directly, through expressions over other nodes and variables, through user-provided functions or through statements in a high-level scripting language. Expressions, functions and scripts introduce the requirement that each node and each variable has a name. Names can be automatically generated but it turns out that naming and name re-factoring are critical aspects of model building. Variables may have nodes as their values and this introduces the need for *pathnames* from a node recursively through nodes held in its variables. A pathname from a node may trace *upstream* through antecedent nodes or *downstream* through consequent nodes in any combination. A node or variable thus may have multiple names either as a global name (nodes only) or a path beginning at some node. The hierarchy of direct identification, expressions, functions and scripting was not an *a priori* design decision. It emerged through sustained conversation with the SmartGeometry group (and others) which established the need for a learning scaffold from the GUI to a full procedural language. Designers move up this scaffold progressively, using new features as both need and skill develop. Each step in the hierarchy necessarily has its own interface.

Expressions and their attendant pathnames introduce a need to understand and interact with the structure of a graph. This is typically supported with *name completion* (not described here) and a *symbolic graph view*.

## 5.3 Symbolic Graph View

A symbolic graph view presents a 2D or 3D visualization of the graph. Its chief uses are to select nodes for editing and as arguments to bind other nodes and variables, to explain the structure and expected behaviour of a graph, and as an aid to debugging. It is also important in selecting subgraphs for compilation into a node (see Section 4.1 above). Nodes may be represented in the symbolic graph view and not in a corresponding 3D interactive view, which makes the symbolic graph view the sole locus for interactive selection of such objects.

The layout of a symbolic graph greatly affects its utility. In a strictly parametric modeller the fact of data propagation establishes the useful convention that a graph may be laid out so that the visual flow never reverses. This is not enough: Figure 5 shows that a well laid out graph can add information critical to understanding a model. Such is important in reuse, which has subtasks of examining and editing graph structure. *Good* layout is task- and model-specific and a matter for authorial intent. Symbolic graph views have specific and limited utility yet occupy a considerable portion of available screen space.



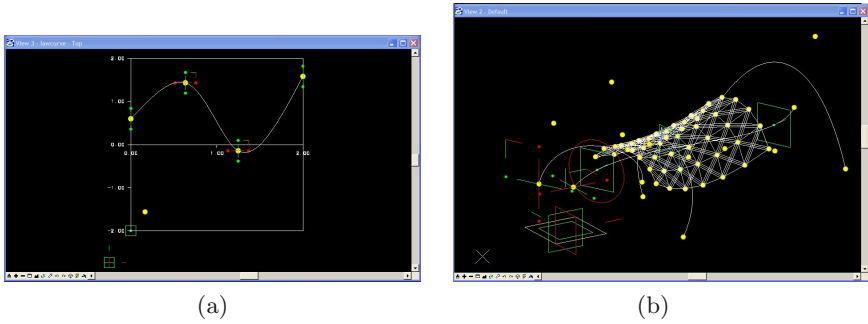
**Fig. 5.** (a) Symbolic graph (left) 3D interaction (right) views in GC. The small window in the centre is the general interface to GC. An order 4 Bezier curve implemented with the deCasteljau algorithm. The symbolic model has four conceptual parts: the *base coordinate system* at the top, the *slider control* on the left, the *systolic array* in the middle and a single node *Bezier curve* on the right. Each has a corresponding 3D display. As the point on the slider is moved from left to right, the point defining the Bezier curve traces out the curve. The nodes in the symbolic graph view have been manually laid out to correspond with both the deCasteljau algorithm and the layout of the control points in the 3D view. (b) A symbolic graph view of the Bezier curve graph after it has been compiled to a single node. The inputs are preserved in the form of four points and a single global variable for the curve parameter. (c) A symbolic graph view in which relatively minor changes to the layout have been made, resulting in a significant degradation of clarity.

## 6 Emergent Features

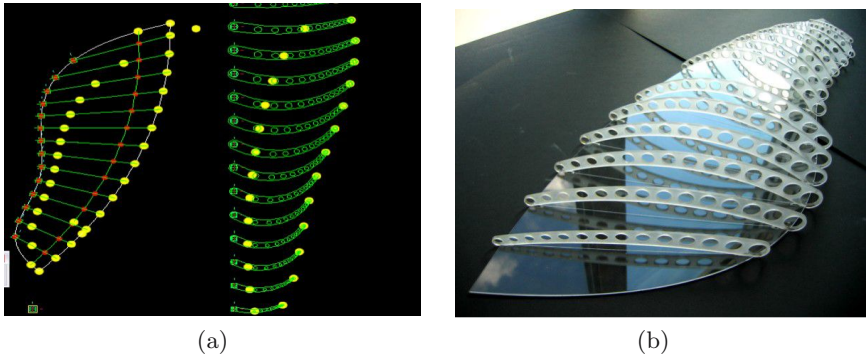
The fact of independent variables that can be directly manipulated in an interface affords the emergent feature that a parametric design system can, to some extent, be its own interface. It turns out that users often spend a considerable portion of their effort building graphs that control other graphs. Such patterns of use have been the impetus to develop generic node types that are, in effect, elements of the GC user interface.

### 6.1 Law Curves

Figure 6 demonstrate the so-called *law curves* in GC that support the mapping from an *independent control variable* to a *dependent control variable*. The law



**Fig. 6.** (a) A *law curve frame*. The ordinate values of the curve become inputs for the parametric model shown in (b).



**Fig. 7.** Wilson Chang is a graduate of the Master's program in Interactive Arts and Technology at Simon Fraser University and is currently (Spring 2005) in private practice. (a) A model comprising a layout for a structural system and the collection of structural members laid out on a plane. The structural members are parameterized by the layout. (b) A physical model whose components have been laser-cut using the structural member descriptions. (*by permission of the author*).

curve frame is nothing more than a node in the graph. It provides a curve controlled by independent variables and a collection of independent control variables that specify abscissa values at which the curve is to be sampled. Its output is an equal-sized collection of dependent control variables that specify the respective ordinate values of the control curve.

## 6.2 Fabrication Planning

As mentioned in Section 1, a principal enabler for parametric design in practice has been the development of fabrication technology by which models, prototypes and entire components can be created by computer-controlled machinery (CNC). Fabrication planning can be brought into a parametric design system by nodes and graphs that transform a design to objects suitable for input to a CNC machine. The shape of objects to be fabricated need not be separate from the

design process, but can become an aspect of the interaction with a model. Figure 7 shows windows to a GC model where a specialized node computes the layout for fabrication and an illustration of the resulting laser-cut model.

## 7 Summary

The advantage of using Generative Components is that it helps me think about what I am doing. The disadvantage is that it forces me to so think. – *Lars Hesselgren*

The structure of design work using parametric systems remains poorly understood. Designers must simultaneously attend to both the specific, concrete design instance and the graph structure that captures its conceptual and mathematical structure. At the same time they must attend to the multifaceted design task at hand. We should neither under-estimate nor under-value the change to the structure of work and design process required. It is crucial to recognize that nothing can be created in a parametric system for which the designer has not explicitly externalized the relevant conceptual and constructive structure. This runs counter to the often-deliberate cultivation of ambiguity that appears to be part of healthy design processes. Abstract concepts such as *intensionality* and *compilation* appear to be essential to effective use. Multiple views appear to be necessary. Such concepts, their attendant tasks and the complexity of working across views provide benefits and impose cognitive costs. Some of these costs may be overcome by clever graphical interventions such as guided layout of a symbolic model. However, any such move must be tested against a robust user community – it far too easy to work on a sophisticated solution to an unimportant problem. In contrast, problems such as the need for a hierarchy from the GUI to the algorithm demand both long conversation with sophisticated designer/users and apt solutions to the appropriate problems that emerge.

## References

- [Bor81] A. Borning. The programming language aspects of thinglab, a constraint-oriented simulation laboratory. *ACM Transactions on Programming Languages and Systems*, 3(4):353–387, Oct. 1981.
- [Gos83] J. Gosling. *The Algebraic Manipulation of Constraints*. PhD thesis, Computer Science Department, Carnegie-Mellon University, May 1983.
- [PMW93] P. Piela, R. McKelvey, and A. Westerberg. An introduction to the ascend modeling system: It's language and interactive environment. *Journal of Magagement information system*, 9(3):91–121, 1993.
- [SMFBB93] M. Sannella, J. Maloney, B. N. Freeman-Benson, and A. Borning. Multi-way versus one-way constraints in user interfaces: Experience with the deltablue algorithm. *Softw., Pract. Exper.*, 23(5):529–566, 1993.
- [Sut63] I.E. Sutherland. Sketchpad: A man-machine graphical communication system. Technical Report 296, MIT Lincoln Lab., 1963.

# Intuitive Shape Modeling by Shading Design

Bertrand Kerautret<sup>1</sup>, Xavier Granier<sup>2</sup>, and Achille Braquelaire<sup>1</sup>

<sup>1</sup> LaBRI, UMR 5800, Université Bordeaux 1; 351, cours de la Libération,  
33405 Talence, France

<sup>2</sup> IPARLA project (INRIA futurs - LaBRI)  
{kerautret, granier, achille}@labri.fr

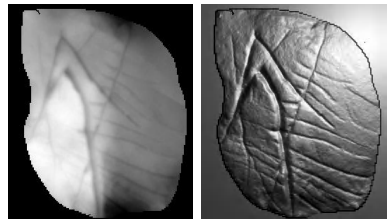
**Abstract.** Shading has a great impact to the human perception of 3D objects. Thus, in order to create or to deform a 3D object, it seems natural to manipulate its perceived shading. This paper presents a new solution for the software implementation of this idea. Our approach is based on the ability of a user to coarsely draw a shading, under different lighting directions. With this intuitive process, users can create or edit a height field (locally or globally), that will correspond to the drawn shading values. Moreover, we present the possibility to edit the shading intensity by means of a specular reflectance model.

**Keywords:** Shape Modeling, Image Based Modeling, Shape From Shading.

## 1 Introduction

The shape of a virtual or real 3D object is mainly perceived by its shading. Our brain is able to infer shape, with only a few ambiguities, from shading alone [18,1]. In drawings and paintings artists use shading to suggest shape. Thus, it seems natural to develop an approach based on shading for modeling and editing 3D objects.

Using painting and drawing techniques is not new in computer graphics. Sketching approaches (e.g. [28,10]) have already explored different solutions as fast and intuitive interfaces for creating and editing 3D models. These solutions have shown that a shape can be reconstructed from a set of strokes, lines (e.g. [28]), or from a defined gesture grammar (e.g. [10]).



In our new approach, we want to achieve a different interaction. The user draws what the final object should look like under different lighting conditions. We believe that it is more intuitive to manipulate a height field by its shading rather than directly the height (see figure above). In this modeling approach, the visual feedback is not provided by the direct visualization of the shape, but by a classical 2D drawing/painting. This process is a direct transfer from the visualization of an object in an artist's mind to the drawing. Then, a height field

is inferred from these shading images, with a minimal difference of the drawn shading.

In this paper, we present the following contributions: (i) an intuitive approach for height field modeling, based on drawn shading images, that is not restricted to integrable shadings; (ii) some editing tools, based on shading editing; (iii) some enhancement tools, based on shading editing. Note that we generate only some height fields. However, these kind of models can still be very useful for, material replacement in image [6], or adding relief to planar surfaces [11], to name just a few.

This paper is organized as follows: after an overview of previous work of shape from shading and sketching interfaces, we briefly present our solution for shape recovery from shading images. Afterwards, we develop our approach for the creation of new height fields and editing of existing ones. Finally, we present some results and applications of this work before we conclude.

## 2 Previous Work

Many techniques are used by artists in order to suggest the object's shape, like characteristic lines (or contour lines) or well-designed shading. Since drawing is a familiar task for a lot of people, many researchers have developed some tools inspired by these drawing/painting metaphors, mostly known as "sketching". These approaches can be classified into two categories: the line or stroke based techniques on the one hand, and the gesture based on the other hand.

The most common approach is to infer a 3D volume from a set of characteristic lines drawn by a user [5,24,25]. This can also be done interactively [17,20]. The ambiguities can be removed by a user-selection of an object list that corresponds to the drawing [16,7]. But these approaches are limited to polyhedral models or pre-defined shapes. With other methods, users can design more complex 3D lines, but the final model is still limited to a wireframe one [22]. On the other hand, the stroke-based approaches allow a larger range of possible 3D lines and 3D curves [21,2]. They are mostly limited to illustration since they cannot really reconstruct a full 3D object.

On the other hand, gesture-based approaches can create a larger range of shapes [28,10,15]. Such methods are based on a gesture grammar which is interpreted to define a contour extrusion. These methods allow a global control on the resulting shape, however, a local control is more difficult.

As the shading mostly depends on the normal (i.e., the local variation of the surface), it seems natural to use it for a better control on the resulting shape. Extending the work of Williams [26], Overveld introduced a shape modeling method by painting the surface gradient [23]. Some constraints guarantee that a solution exists, but they disallow the use of classical 2D paint programs and thus reduce the user's freedom. An original approach to edit shape by using shading information was introduced by Rushmeier et al. [19]. This method was used to restore the original shape by using a shading image associated with the region to be edited. This approach is convenient to edit small parts of an existing 3D shape, but it is not adapted to create a new shape without an initial 3D model.



Moreover, this approach can suffer from the limitation of the shading intensity contained in only one shading image.

The first method to reconstruct shape from shading intensity was introduced by Horn in the 70's years [8]. From this first approach, a variety of different methods appeared, but as shown in the survey from Tsang et al. [30], these methods are not robust, especially for the reconstruction from real images. This low robustness represents an important drawback to envisage the shape reconstruction only from the shading information of the drawings.

Even if we suppose a robust shape from shading method, the following points are always problematic:

- A first reason is the well known concave/convex ambiguity [18,1], which becomes particularly apparent when the light source is located in the viewing direction. This ambiguity implies the use of an initial guess in order to recover the shape in such a light source direction.
- When the light source is not in the viewing direction, there is the presence of shadows or inter-reflection. As a consequence, there are image areas which do not contain valid shading information.
- If we consider a drawing, the shading intensity could correspond to an impossible shading image [9]. For example, a compact dark blotch on unit brightness background cannot be a shaded image of a surface with continuous first derivatives.
- A large number of methods first recover normal orientations and then integrate them in order to recover the shape of the final surface. In the presence of noise, the final shape depends on the integration method.

A possible alternative to avoid the problem of a non unique solution of the shape from shading problem is to use more shading information by considering several images associated to different light source directions. This approach, called photometric stereo, was introduced by Woodham [27]. It provides a unique solution for the surface orientation. The results obtained from real data are degraded by noise, shadows, or integrability of the surface normal. There are several approaches that used  $N$  light source directions, as for example in [29], but the main drawback is the requirement of a large number of images in order to obtain a precise reconstruction.

### 3 Shape Reconstruction from Shading Intensity

Since our goal is the reconstruction of a shape from one or several shading images which can differ from real shading intensity, we need a really robust reconstruction method. Classical shape from shading or photometric stereo methods are often not robust both in terms of noise and photometric information. For all these reasons, we used a robust approach which shows precise reconstructions with a few number of images [3].

### 3.1 Reconstruction by Equal Height Region Propagation

The reconstruction method introduced in [3,12] is based on the equal height region propagation. The strategy of reconstruction consists in considering different contours of equal height in order to reduce the number of solutions. Generally, it is convenient to define the surface normal orientation by the partial derivatives  $p$  and  $q$  of the surface that is defined as the bivariate elevation function  $Z(x, y)$ . Another possibility is to consider the decomposition of the normal vector by the azimuth ( $\phi$ ) and polar angle ( $\theta$ ). Now, if we assume that the horizontal component ( $\phi_0$ ) of the normal vector is known, it can be shown that for a Lambertian surface the solution for the vertical orientation of the normal vector is reduced to the two following solutions:

$$\theta = \arctan \left( \frac{-\omega \pm \sqrt{-K^4 + K^2 + \omega^2 K^2}}{K^2 - \omega^2} \right), \quad (1)$$

with  $K = L_i \sqrt{1 + p_s^2 + q_s^2}$ ,  $\omega = p_s \cos \phi + q_s \sin \phi$  and  $p_s, q_s$  are associated to the light source direction.  $L_i$  represents the image intensity.

There are several strategies to select one of the two possible solutions. For example, some continuity constraints can be applied according to the neighborhood, but a more robust approach consists in using more than one shading image in order to overcome this ambiguity.

The reconstruction principle is based on the evolution of patches initialized according to the plane areas of the source images which are orthogonal to the viewer direction. From these initial regions, the reconstruction is made by directly estimating the height of the points that are connected to the patch. In a second step, after this height estimation, the patches are updated by adding all the points for which their new height is included in the equal height interval. This condition is used in order to keep the frontier points of the patch at the same height and thus limiting the possible number of solutions as shown previously.

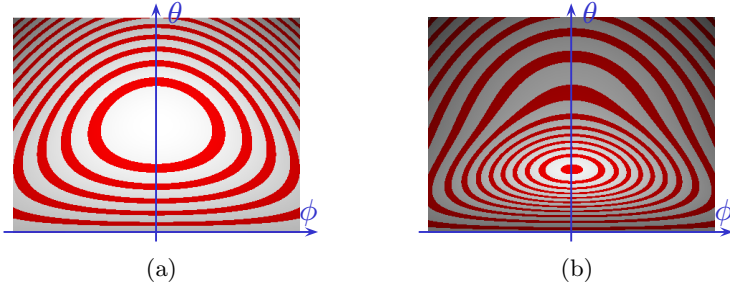
More details about this height estimation and the patch evolution can be found in [3].

### 3.2 Extension to Specular Models

Since the height estimation process is not explicitly based on the mathematical expression of the Lambertian surface, this method can be extended to other reflection models such as the extended Lambertian model or the specular/Lambertian model. For the reconstruction of glossy surfaces, we have used the hybrid specular/Lambertian model based on the model of Nayar et al. [14] containing two terms associated to the diffuse and specular component:

$$I = \rho_{ld} \max(0, \cos(\theta_i)) + \rho_{ls} e^{\frac{\alpha^2}{2\sigma}} \quad (2)$$

where  $\alpha$  is the angle between the normal and the bisector vector of the incident light source and the viewer direction.  $\rho_{ld}$  and  $\rho_{ls}$  represent the coefficient of the



**Fig. 1.** Comparison of the reflectance map of a Lambertian (a) and a specular surface (b). The iso-intensity contours are represented in dark.

diffuse and specular component respectively, and  $\sigma$  is associated to the roughness surface. Note that the specular spike was not considered.

Generally the Lambertian model associated to a matt surface seems to be well adapted to intuitively draw or complete an existing source image. This model presents the advantage to have an intensity distribution which appears well adapted according to the normal of the surface. Such a distribution can be visualized on the reflectance map which is usually used to represent the distribution of the reflected intensity with respect to the normal orientation  $(\phi, \theta)$ . An example of a reflectance map associated to a Lambertian surface is represented in Fig. 1-(a). Now, if we consider the reflectance map of glossy/specular models, it appears that the intensity is concentrated in a region associated to the normal orientation near the direction of the light source. This distribution is well visible on the reflectance map in Fig. 1-(b).

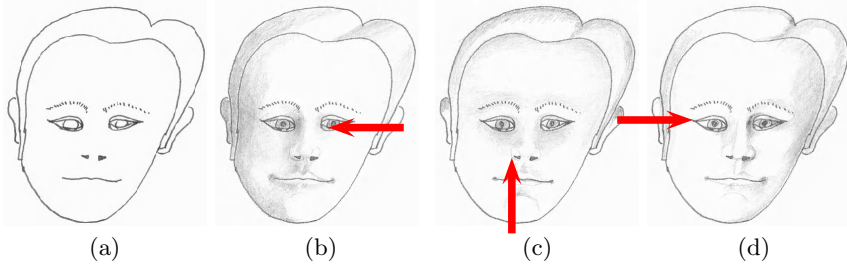
From these comparisons of different types of reflectance maps, one can think that it is more difficult for a user to draw fine details with a Lambertian reflectance model since it implies a large precision in using the different shading intensities. On the other hand, with the specular reflectance, fine details could be easier to draw since the intensity gradient is much more important in a particular direction.

## 4 Creation, Editing and Enhancement of Height Fields

Generally, before the final drawing of an object or a character, an artist draws multiple sketches for the different viewing directions and the different lighting conditions. We use this technique for the development of our new modeling and editing approach.

### 4.1 Creating the Height Field

The first step in the creation of a new height field is the definition of the local and global contours. These can be directly drawn (see Fig. 2-(a)) or extracted from existing images (see Fig. 5-(b)). This step plays an important role since it could help the user to locate the main features in the different drawings associated



**Fig. 2.** Generating source images (b-d) from an initial contour (a). The Images (b-d) are the three of the four images used for the reconstruction. The shading has been drawn according to the light source direction with a vertical orientation of  $45^\circ$  and a horizontal orientation defined by the arrow.

to different light source directions. These images are the support of the shading definition and contain some discontinuities that we want to keep in the final reconstruction.

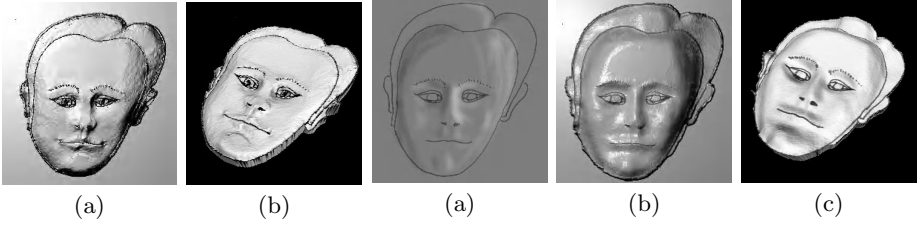
In the second step, the user can draw the different shading information associated to several predefined light source directions (see Fig. 2). The orientation of the light source direction can be chosen by the user according to different intervals. Note that this step can be done using a classical drawing software, or directly on a real paper by using a scanner, thanks to our robust reconstruction.

The final step is the reconstruction itself. As the reconstruction is done off-line, the whole process is not interactive compared to interactive height field reconstruction, like in [23]. But during the whole process, the user has a direct feedback of what the final height field would look like by means of the shading that is currently drawn.

## 4.2 Height Field Enhancement and Completion

We can also use a similar idea to enhance an existing height field or the result of the previous reconstruction. Since a height field can be represented as a gray-scaled image, all 2D filters can be directly applied. But, once again, the direct manipulation of height values is less intuitive than the manipulation of shading values. We propose here a three step process. First, we extract some shading images from the existing height field for several light directions. Note that this step is not needed when we already have shading images, like those obtained in the previous section. Then, these images are filtered locally or globally and shape from shading is performed on these new shading images.

We also propose to use a similar approach to add the missing part of an existing object. As for the creation, we can use 2D painting tools to achieve this task. This again is a three step process. We first extract shading images for different light directions. As for the height field enhancement, this step is not needed if we already have the shading images. In the second step, the missing parts can be manually added by using for example the clone-brush of image editing software, or an automatic process like in [4]. We can use all the power of



**Fig. 3.** Reconstruction results obtained from the drawings of faces (image (a)). The image (b) was obtained after rendering the resulting surface and (c) is the 3D visualization with the original contour of Fig. 2-(a) used to create the source image (a).

2D painting/drawing approaches for completing the missing part of an existing object. The last step is also the final reconstruction.

Compared to mesh-based solution like in [13], we are more restricted (only a 2.5D object), but our approach gives the complete freedom to the user for drawing the missing part, by exploiting all its expertise and knowledge about the edited model. We also believe that it is more intuitive to complete the missing shading than the missing height field. Moreover, it can be done by a non computer experienced user, independently of a specified modeling tool.

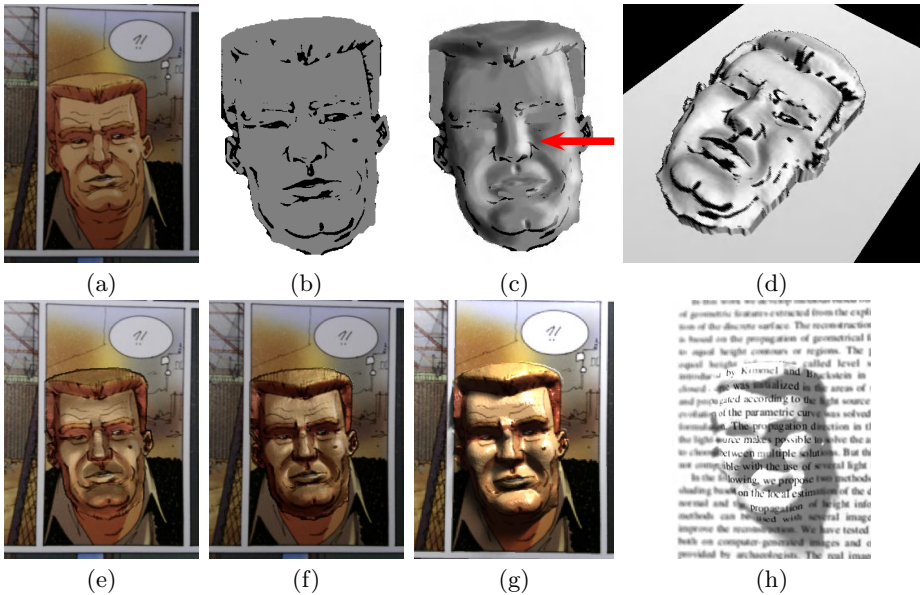
## 5 Results

In this section, we present reconstruction results of different types of drawings, the application for height field enhancement, as well as original applications.

### 5.1 Reconstruction from Drawing Patterns

From the previous drawings of Fig. 2, we applied the reconstruction with the four source images. The reflectance model used for the reconstruction was the Lambertian model for which the albedo was estimated automatically according to the patch initialization on plane areas (i.e., the front and the cheeks of the face). The resulting height field was initially flat and the height interval was adjusted in order to obtained a better visual effect on the resulting surface. Fig. 3-(a-b) illustrates the final result of the reconstruction. The image (a) was obtained after rendering the resulting height field with a non-distant light source and by a combined specular/Lambertian model. The initial contour was superimposed with the rendered image. In the same way, the image (b) of the Fig. 3 shows the 3D visualization of the reconstructed surface.

From the same contour used to create the previous drawings, we have generated four coarse shading images using a classical drawing software. The image (a) of Fig. 4 shows one of the four images used for the reconstruction. These images were first filled by the shading intensity corresponding to plane areas according to the light source direction. Then, different areas were whitened or



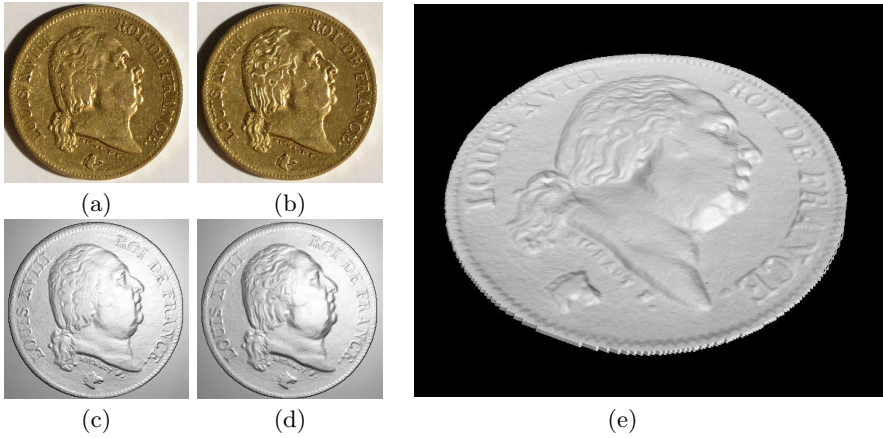
**Fig. 5.** Generating sources images (c) from an existing drawing (a-b). The resulting shape is shown in (d). The images (e-h) illustrate the application to image rendering. ©Brahya/Corbeyran/Braquelaire–Dargaud-Bnlux (s.a. Dargaud-Lombard n.v.)-2004.

blackened in an intuitive way. The resulting surface presents a different aspect than the previous example and the final result perfectly fits the initial contour that is superimposed to the resulting surface (Fig. 4-(b-c)).

## 5.2 Application to Image Rendering

These reconstruction results can be used for original applications. A possible application is to use the resulting height field in order to improve the original drawing from the initial image. For example, it is possible to obtain new illumination effects by adding a light source including shading areas and shadows defined from the height field. The source image was used as a texture that is mapped to the associated height field. Fig. 5 illustrates such obtained new rendering effects. The initial image is shown in Fig. 5-(a) and the new images obtained with different light source directions and different material properties is illustrated in the images (e-g). The resulting images appear still similar to the source images, but they look more realistic than the original. Thanks to the height field which allows to use a non-distant light source rendering, using not only the normals of the surface, but also the height of each point of the surface.

Another application consists in applying blurring effects on the original source image. Since not only the normals of the surface are available, it is possible to simulate a focal length on the original drawing. The image (h) of Fig. 5 illustrates this effect.



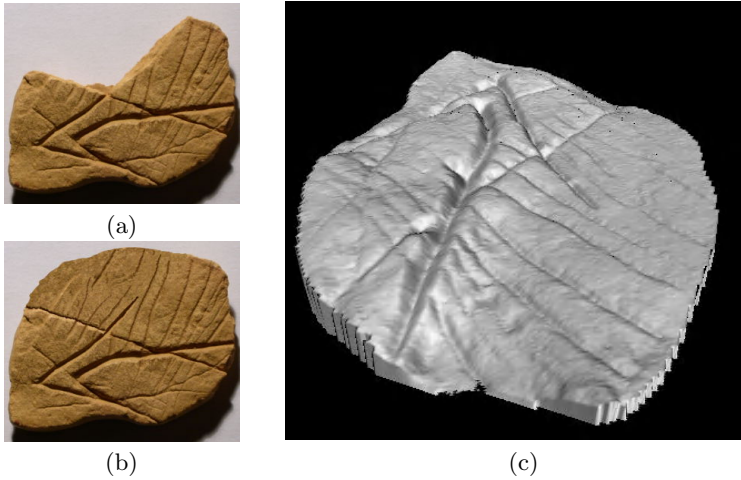
**Fig. 6.** Reconstruction results from image enhancement of specular object. The image (a) represents one of the four images of the original object. The image (b) was obtained after adding details on the original image (a). Images (c-e) illustrate the reconstruction obtained from the previous images.

### 5.3 Image Restoration for Shape Reconstruction

In order to measure the possibility to use this approach to restore and reconstruct shape, we have selected different old objects. A first example was an archaeological object aged about 35,000 years. This object presents a missing part which was broken (Fig. 7-(a)). In order to reconstruct the object entirely with the missing part, we have completed the image by using the clone brush tools of an image editing software (image (b)). The details of the missing part have been drawn both by continuity and by applying archaeological criteria. Three images were used for the reconstruction. The time to complete the source images was not so much important and was around 20-30 minutes for each image. The resulting surface presented in the Fig. 7-(c) shows as if it was the original object, and there is no difference between the existing and the virtual part of the shape.

Finally, we have applied this main idea to a non-Lambertian object for the reconstruction of an old coin dating from 1818. The coin was used meanwhile and a number of details were smoothed out or disappeared in comparison to a better conserved coin (Fig. 6-(a)). In order to recover the aspect of the original coin we have edited the four source images used for the reconstruction. It was made by drawing details as the hair, or by accenting the contrast around the letter contours which were smoothed into the original images. From the restored images, we have applied the reconstruction, and all the details performed by the drawing were well reconstructed and visible on the rendering of the resulting height field (Fig. 6-(d)). The different coefficients of the Lambertian/Specular model from Eq. 2 were approximatively estimated by a roughness coefficient  $\sigma = 0.3$  and by a diffuse/specular constant  $\rho_{ts} = 0.7$ ,  $\rho_{ld} = 0.3$ .

The reconstructions were experimented on an *Intel Pentium 3 1GHz* with 256Mb of memory. The average times of the different reconstruction processes were included between 2 and less than 5 minutes for a resolution of  $500 \times 488$



**Fig. 7.** Illustration of result obtained from image completions. The image (b) was obtained after completing the image (a). The reconstruction result after the completion of the three source images is shown in (c).

(resolution of the coin of Fig. 6). The reconstruction process was not optimized since the reconstruction can be done offline after all the drawings were made. Note that the reconstruction process can be adapted to the partial reconstruction of the surface when we consider an interactive surface reconstruction.

## 6 Conclusion

In this paper, we have presented an original method for modeling height fields from shading images. This approach, based on a robust reconstruction method, allows the construction of a shape from shading that are manually drawn by a user. We have presented three possible applications of such an approach, from height field creation, to height field completion and enhancement. The creation or editing process presented here consists always of three steps: users create reference images, and then edit or draw the shading for different lighting directions. The final step is a shape from shading reconstruction. Even if this work is still limited to height fields, it has shown that working on shading is an intuitive approach for modeling shapes.

This work brings up several fields of reflection. First, we want to create a fully integrated tool that enhances the user-interaction. We also want to experiment more 2D tools as modeling solutions through shading editing/creation. Finally, we consider to apply this reconstruction to full 3D object reconstruction and editing.

## Acknowledgments

We thank Francesco d'Errico from the *Institut de Préhistoire et Géologie du Quaternaire* for having kindly provided the numerical data of archaeological



objects used in this work. We also thank Patrick Reuter for numerous language corrections.

## References

1. E.H. Adelson and A.P. Pentland. *Perception as Bayesian Inference*, chapter The Perception of Shading and Reflectance, pages 409–423. Cambridge University Press, September 1996.
2. D. Bourguignon, M.-P. Cani, and G. Drettakis. Drawing for illustration and annotation in 3D. *Computer Graphics Forum (Proc. of Eurographics 2001)*, 20(3):114–122, September 2001.
3. A. Braquelaire and B. Kerautret. Reconstruction of lambertian surfaces by discrete equal height contours and regions propagation. *Image and Vision Computing Elsevier*, 23(2):177–189, February 2005.
4. I. Drori, D. Cohen-Or, and H. Yeshurun. Fragment-based image completion. *ACM Trans. Graph. (Proc. SIGGRAPH 2003)*, 22(3):303–312, 2003.
5. L. Eggli, C.-Y. Hsu, B.D. Brderlin, and G. Elber. Inferring 3D models from free-hand sketches and constraints. *Computer-Aided Design (JCAD)*, 29(2):101–112, February 1997.
6. H. Fang and J.C. Hart. Textureshop: texture synthesis as a photograph editing tool. volume 23, pages 354–359. ACM Press, 2004.
7. M.J. Fonseca, A. Ferreira, and J.A. Jorge. Towards 3d modeling using sketches and retrieval. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling*, August 2004.
8. B.K.P. Horn. *Shape from Shading: a Method for Obtaining the Shape of a Smooth Opaque Object from One View*. PhD thesis, Departement of Electrical Engineering, MIT, 1970.
9. B.K.P. Horn, R.S. Szeliski, and A.L. Yuille. Impossible shaded images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(2):166–170, 1993.
10. T. Igarashi, S. Matsuoka, and H. Tanaka. Teddy: a sketching interface for 3d freeform design. In *SIGGRAPH '99: Proc. of the 26th annual conference on Computer graphics and interactive techniques*, pages 409–416. ACM Press/Addison-Wesley Publishing Co., 1999.
11. T. Kaneko, T. Takahei, M. Inami, N. Kawakami, Y. Yanagida, T. Maeda, and S. Tachi. Detailed shape representation with parallax mapping. In *Proc. of the ICAT 2001 (The 11th International Conference on Artificial Reality and Telexistence)*, pages 205–208, December 2001.
12. B. Kerautret. A robust discrete approach for shape from shading and photometric stereo. In *ICCVG*, pages 581–587, Varsawa, Poland, 2004. Springer-Verlag.
13. B. Levy. Dual domain extrapolation. *ACM Trans. Graph. (Proc. of SIGGRAPH 2003)*, 22(3):364–369, July 2003.
14. S.K. Nayar, K. Ikeuchi, and T. Kanade. Surface reflection: physical and geometrical perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):611 – 634, 1991.
15. S. Ohwada, F. N., K. Nakazawa, and T. Igarashi. A sketching interface for modeling the internal structures of 3d shapes. In *Proc. International Symposium on Smart Graphics*, pages 49–57. Springer-Verlag LNCS 2733, July 2003.
16. J.P. Pereira, V.A. Branco, J.A. Jorge, N. F. Silva, T.D. Cardoso, and F.N. Ferreira. Cascading recognizers for ambiguous calligraphic interaction. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling*, August 2004.

17. D. Pugh. Designing solid objects using interactive sketch interpretation. In *Proc. Symposium on Interactive 3D graphics*, pages 117–126. ACM Press, 1992.
18. V.S. Ramachandran. Perceiving shape from shading. *Scientific American*, 259(2):76–83, August 1988.
19. H. Rushmeir, J. Gomes, L. Balmelli, F. Bernardi, and G. Taubin. Image-based object editing. In *Fourth International Conference on 3-D Digital Imaging and Modeling (3DIM '03)*, pages 20–28, October 2003.
20. A. Shesh and B. Chen. Smartpaper—an interactive and user-friendly sketching system. *Computer Graphics Forum (Proc. of Eurographics 2004)*, September 2004.
21. O. Tolba, J. Dorsey, and L. McMillan. Sketching with projective 2D strokes. In *UIST '99: Proc. of the 12th annual ACM symposium on User interface software and technology*, pages 149–157. ACM Press, 1999.
22. S. Tsang, R. Balakrishnan, K. Singh, and A. Ranjan. A suggestive interface for image guided 3d sketching. In *ACM CHI Letters (ACM CHI Conference on Human Factors in Computing Systems)*, volume 6, pages 591–598.
23. C.W.A.M. van Overveld. Painting gradients: free-form surface design using shading patterns. In *Proc. Graphics interface '96*, pages 151–158. Canadian Information Processing Society, 1996.
24. P.A.C. Varley, H. Suzuki, and R.R. Martin. Can machines interpret line drawings? In *Eurographics Workshop on Sketch-Based Interfaces and Modeling*, August 2004.
25. P.A.C. Varley, Y. Takahashi, J. Mitani, and H. Suzuki. A two-stage approach for interpreting line drawings of curved objects. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling*, August 2004.
26. L. Williams. 3D paint. In *Proc. SI3D '90*, pages 225–233. ACM Press, 1990.
27. R. J. Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineering*, 19:139–144, 1980.
28. R.C. Zeleznik, K.P. Herndon, and J.F. Hughes. SKETCH: an interface for sketching 3D scenes. In *Proc. SIGGRAPH '96*, pages 163–170. ACM Press, July, 1996.
29. R. Zhang and M. Shah. Iterative shape recovery from multiple images. *Image and Vision Computing*, pages 801–814, 1997.
30. R. Zhang, P.-S. Tsai, J.E. Cryer, and M. Shah. Shape from shading: A survey. *IEEE PAMI*, 21(8):690–706, August 1999.

# Automatic Cross-Sectioning Based on Topological Volume Skeletonization

Yuki Mori<sup>1</sup>, Shigeo Takahashi<sup>2</sup>, Takeo Igarashi<sup>3</sup>, Yuriko Takeshima<sup>4</sup>,  
and Issei Fujishiro<sup>4</sup>

<sup>1</sup> Department of Computer Science, The University of Tokyo,  
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8654, Japan  
yuki@ui.is.s.u-tokyo.ac.jp

<sup>2</sup> Graduate School of Frontier Sciences, The University of Tokyo,  
5-1-5 Kashiwanoha, Kashiwa, Chiba 277-8561, Japan  
takahashis@acm.org

<sup>3</sup> Department of Computer Science, The University of Tokyo / PRESTO,  
JST 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8654, Japan  
takeo@acm.org

<sup>4</sup> Institute of Fluid Science, Tohoku University,  
2-1-1 Katahira, Aoba-ku, Sendai 980-8577, Japan  
{takeshima, fuji}@vis.ifs.tohoku.ac.jp

**Abstract.** Cross-sectioning is a popular method for visualizing the complicated inner structures of three-dimensional volume datasets. However, the process is usually manual, meaning that a user must manually specify the cross-section's location using a repeated trial-and-error process. To find the best cross-sections, this method requires that a user is knowledgeable and experienced. This paper proposes a method for automatically generating characteristic cross-sections from a given volume dataset. The application of a volume skeleton tree (VST), which is a graph that delineates the topological structure of a three-dimensional volume, facilitates the automated generation of cross-sections giving good representations of the topological characteristics of a dataset. The feasibility of the proposed method is demonstrated using several examples.

## 1 Introduction

The visualization of volume datasets is important in many scientific fields, from geophysics to biomedical sciences. Researchers have proposed a number of visualization techniques, including volume rendering and isosurface extraction, but it is still difficult to comprehend the complicated inner structures of volume datasets.

Volume rendering is one of the most commonly used visualization techniques, in which a transfer function (TF) determines how the volume dataset is rendered. The design of the TF, which assigns opacity and color values to each voxel in the data, is crucial and has become a significant research topic. Researchers have proposed many different semi-automatic transfer function design methods, which can be divided into two major categories: image-centric [1, 2] and data-centric [3-6]. An image-centric

method presents a broad selection of TFs and generates corresponding visualization images so that the user can choose the best one [1, 2]. However, these methods do not consider the meaning (*i.e.*, significant features) of target volume datasets. In contrast to image-centric methods, a data-centric method tries to find the best transfer function by rigorously analyzing these features of the input volume datasets [3-6].

Isosurface extraction is another commonly used visualization technique that renders a three-dimensional surface corresponding to points with a single scalar value. Lyness and Blake presented real time isosurface browsing [7], and Itoh and Koyamada presented a method for automatic isosurface propagation [8]. However, both volume rendering and isosurface extraction techniques are still limited in their ability to help researchers visualize detailed structures, because two-dimensional projection of three-dimensional volumetric information always involves some occlusions.

Therefore, users frequently use cross-sections to inspect the inner structures of a volume dataset. Detailed illustrations in biology textbooks and scientific magazines demonstrate the effectiveness of using cross-sections of a volumetric object to explain internal structures. Many researchers have explored ways to enhance cross-sectioning of volume datasets. Owada *et al.* presented an interactive system for designing and browsing volumetric illustrations [9], with which a user can cut a three-dimensional model and see detailed textures of its internal structure in the cross-section. Hinckley *et al.* discussed a two-handed user interface for three-dimensional neurosurgical visualization [10]; their technique involves generating a cutting plane by operating two devices: one corresponding to the target volume; the other corresponding to the cutting plane. Viegas *et al.* proposed three-dimensional magic lenses [11]; the lens volume is set interactively, changing the visual appearance of the geometry intersecting the lens volume. Diepstraten *et al.* presented ways to map cutaway renderings directly to modern graphics hardware in order to achieve interactive frame rates [12]. All of these existing systems require users to manually search for an ideal cross-section by fine-tuning many parameters; this is tedious and time-consuming. In addition, users may fail to find a cross-section that appropriately illustrates the volume dataset's inner structures.

To address these problems, we present a method for the automatic generation of cross-sections that reveal characteristic structures of a volume dataset. To extract the characteristic structures of a volume dataset, we use an abstract graph called a volume skeleton tree [13], which represents splitting and merging of isosurfaces with a varying scalar field value. Each node of the graph represents a critical point where merging and splitting happens, and each link represents a region between the isosurfaces represented by its end nodes. Volume skeleton trees have been used to perform important operations in analyzing volume datasets, such as extraction of a critical isosurface or representative isosurface [13] and interval volume decomposition (IVD) [14]. Several authors have proposed methods for extracting similar skeletons from volume datasets [15-17]. While these algorithms are computationally elegant, they do not address changes in isosurface topology (*i.e.*, genera) with respect to the scalar field value, and hence cannot maintain topological consistency of extracted features.

Our method is related to other systems that find characteristic views of target objects. Chakravarty and Freeman [18] presented a method to classify all possible views of an object into a set of characteristic views, within which all views are topologically identical. Agrawala *et al.*'s system searches for a view that increase the visibility of the parts to create effective assembly instructions [19].

This paper begins with a description of the algorithm for extracting a volume skeleton tree from a volume dataset. Then, Section 3 describes the algorithm for generating cross-sections of a volume dataset based on the extracted VST and cross-section indication techniques. Section 4 is an application of the methodology. Finally, Section 5 discusses the methodology's extensibility and limitations and addresses related issues for future research.

## 2 The Volume Skeleton Tree

In general, volume data is represented as a three-dimensional single-valued function:

$$w = f(x, y, z) \quad (1)$$

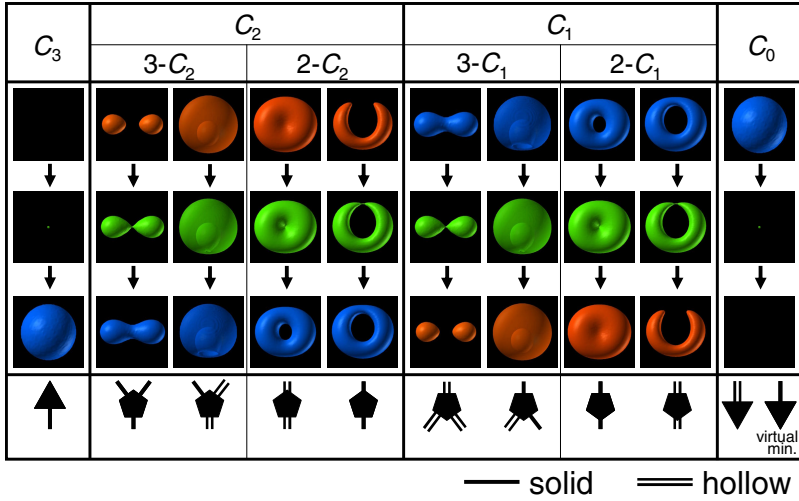
where  $x$ ,  $y$ , and  $z$  represent ordinary three-dimensional coordinates and  $w$  represents the corresponding scalar field value. A volume is continuous when it is defined as an analytic function, but it usually consists of discrete grid samples. In this case, we apply linear interpolation to construct a continuous volume. Given a continuous volume, isosurfaces can be extracted by collecting points with a constant scalar value. By gradually changing the scalar value, a sequence of isosurfaces with varying topological structure can be determined. A volume skeleton tree (VST) [13] represents such global topological changes in the form of a tree.

A VST node represents a *critical point* where either the number of connected components or the genus of the isosurfaces changes when reducing the scalar value. They are classified into four groups: maxima ( $C_3$ ), saddles ( $C_2$ ), saddles ( $C_1$ ), and minima ( $C_0$ ), which represent isosurface appearance, merging, splitting, and disappearance, respectively (Figure 1). A VST link represents an isosurface component that is termed *solid* if it expands and *hollow* if it shrinks; solid links are represented as single lines, and hollow links as double lines. The isosurface merging at  $C_2$  and splitting at  $C_1$  has four topological transition paths with different isosurface spatial configurations. For convenience, there is an assumption that all boundary voxels are connected to the virtual minimum with a scalar field value of  $-\infty$  [13]. Note that when the node is the virtual minimum, the link incident to a  $C_0$  node is solid, as shown in Figure 1. In this model, the node has coordinates and a scalar field value, while the link has the genus and index of adjacent nodes.

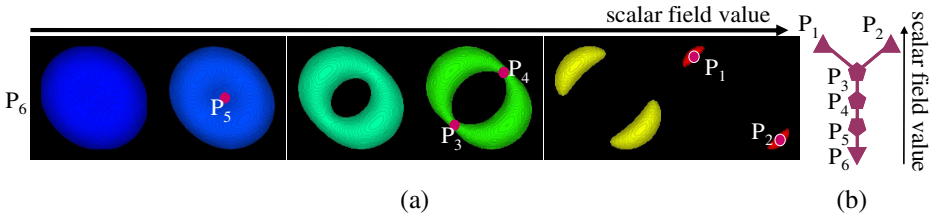
Figure 2(a) shows an isosurface structure of a volume dataset, calculated from the following analytic volume function:

$$\begin{aligned} f(x, y, z) = & 4c^2 ((x-R)^2 + (z-R)^2) \\ & - ((x-R)^2 + y^2 + (z-R)^2 + c^2 - d^2)^2 \\ & + 4c^2 ((x+R)^2 + (z+R)^2) \\ & - ((x+R)^2 + y^2 + (z+R)^2 + c^2 - d^2)^2 \end{aligned} \quad (2)$$

Where  $0 < d < c$ ,  $c^2 + d^2 \geq 6R^2$ .



**Fig. 1.** The connectivity of a critical point of each type in a volume skeleton tree: Single and double lines represent links corresponding to solid and hollow isosurfaces, respectively.



**Fig. 2.** Topological analysis of the analytic volume function (2): (a) A change in isosurface and critical points; (b) The corresponding volume skeleton tree.

Function (2) has six critical points:  $P_1, P_2$  (appearance),  $P_3, P_4$  (merging),  $P_5$  (splitting), and  $P_6$  (disappearance).

As described previously, this algorithm accounts for the virtual minimum; this is artificially added to the volume function (1) so that we can think of an input dataset as a topological three-dimensional sphere  $S^3$ . This enables us to check the mathematical consistency of extracted critical points by consulting the Euler formula [11]:

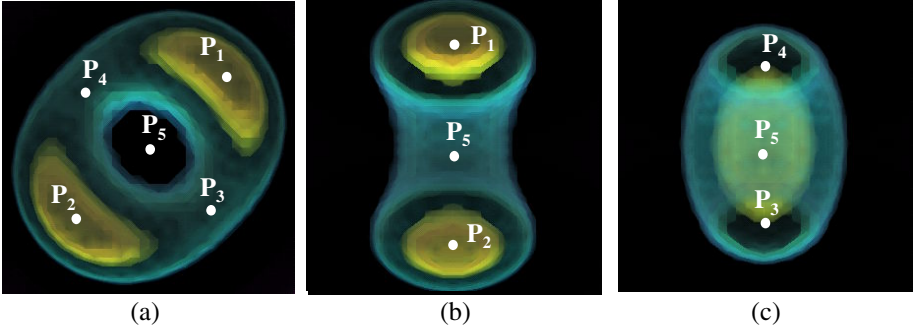
$$\#\{C_3\} - \#\{C_2\} + \#\{C_1\} - \#\{C_0\} = 0 \quad (3)$$

where  $\#\{C_i\}$  represents the number of critical points of index  $i$ .

Although the volume skeleton tree effectively captures the topological skeleton of a volume dataset, it is often affected by high-frequency noise that produces a significant number of minor critical points. Therefore, it is necessary to remove minor critical points and simplify the VST in order to extract an important global skeleton from an unknown volume. This process is currently semi-automatic; the system gradually simplifies the tree until the user satisfies the result and stops the process.

### 3 Generating Cross-Sections

Our basic idea is to use a VST to extract characteristic points from a dataset and then cut the dataset with a plane that best fits these target points. For example, we can consider the volume function (2) introduced in the previous section. Our goal is to extract characteristic cross-sections, such as those shown in Figure 3; The dataset has three axes of symmetry. Figure 3 shows three different cross-sections each of which passes through the dataset's center of gravity and two of these axes. This section describes an algorithm to generate such cross-sections using a VST.



**Fig. 3.** Examples of cross-sections of the analytic function volume (2): (a) A cross-section passing through many critical points; (b), (c) Cross-sections passing through axes of symmetry.

#### 3.1 Computing Cutting Planes Using Volume Skeleton Trees

A simple method to generate a cutting plane is ‘least-squares’ fitting to a group of critical points of  $N$ . We can construct a covariance matrix (4) of the points:

$$A = \frac{1}{N} \sum_i \begin{pmatrix} (x_i - \bar{x})^2 & (x_i - \bar{x})(y_i - \bar{y}) & (x_i - \bar{x})(z_i - \bar{z}) \\ (x_i - \bar{x})(y_i - \bar{y}) & (y_i - \bar{y})^2 & (y_i - \bar{y})(z_i - \bar{z}) \\ (x_i - \bar{x})(z_i - \bar{z}) & (y_i - \bar{y})(z_i - \bar{z}) & (z_i - \bar{z})^2 \end{pmatrix} \quad (4)$$

where  $\bar{x}$ ,  $\bar{y}$  and  $\bar{z}$  are averages of  $x$ ,  $y$  and  $z$ , respectively. Then, we calculate the eigenvalues and eigenvectors. We can then construct three planes, each of which passes through the center of gravity of the point group and includes any two of the three eigenvectors. By default, the system uses the plane that includes two eigenvectors paired with the two major eigenvalues, but the user can also select two other cross-sections. This selectivity is important because the default cross-section is not always the desired one. We are testing two techniques to obtain target points; one focuses on VST nodes (critical points) and the other focuses on VST links (interval volumes).

The first technique uses critical points as a fitting target; this provides cross-sections to reveal points where the isosurface topology is going to change.

The second technique computes the center of gravity of each interval volume and uses them as a fitting target; weight is assigned to points based on the volume size, and used when fitting the plane.

### 3.2 Displaying Cross-Sections

A cross-section is displayed to the user using one of the following three methods:

The first method converts each of the field values in a cross-section into a suitable color value and displays them on screen. This provides a user with detailed information because the user can see the exact value of each point in the cross-section.

The second method polygonizes isosurfaces associated with critical points and cuts them at the given cutting plane; the system then makes the near side of the plane transparent, making internal structures visible. This method is useful for allowing a user to know the cutting plane's location and to operate on isosurfaces (*e.g.*, hiding an isosurface by clicking on it).

The third method volume-renders the dataset on the far side of the cutting plane. This display method does not show the exact values of the cross-section, however can be expected to provide a significant visual effect to be able to embed 2.5 dimensional information in 2D images.

Figure 3 shows that our method can find appropriate cross-sections of function (2). In this example, we used the third method; Figure 3(a) is rendered with the best-fitting plane. Figure 3 (b) and (c) are rendered with the two next-best candidates.

## 4 Application to Real Datasets

We applied our method to three datasets taken from quantum science and medical science in order to demonstrate its applicability to real datasets. In each dataset, scalar field values were normalized to an 8-bit range [0, 255]. The system runs in an ordinary PC environment (CPU: Pentium IV, 1.6 GHz, RAM: 1 GB).

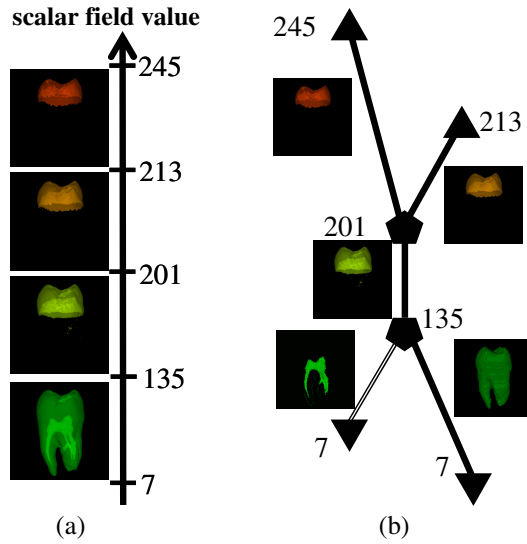
As we described in the algorithm section, we implemented and tested two techniques for computing a cutting plane: one using critical points and the other using interval volumes. However, the resulting cutting planes were almost identical, so we only provide results obtained using the first technique. In future work, we will investigate the differences in more details.

### 4.1 Tooth

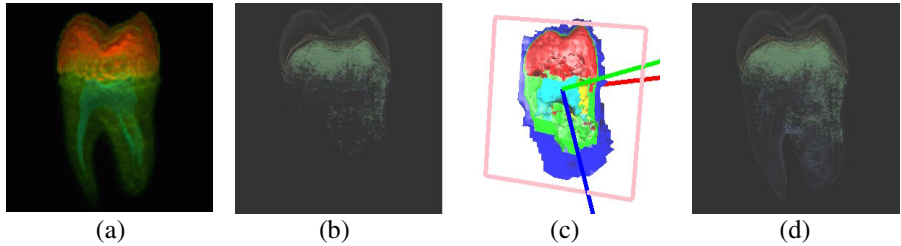
The first target was a medical CT-scanned dataset, hereafter called 'tooth', which consists of  $128 \times 128 \times 80$  voxel data with 8-bit scalar values. The original dataset ( $256 \times 256 \times 161$ , 16-bit) was used as one of three target datasets in an interesting panel in *Visualization 2000* [20, 21]. The dataset suffers from high-frequency noise, and was downsized into  $16 \times 16 \times 10$  voxels to generate the volume skeleton tree.

An automatic analysis with default settings oversimplified the VST, so we interrupted the simplification process along the way, producing six critical points. This effectively allowed us to distinguish the tooth's four anatomical regions (cement, enamel, dentin, and pulp) (Figure 4).





**Fig. 4.** National Library of Medicine (NLM) tooth volume: (a) A change in isosurface and critical points; (b) The corresponding volume skeleton tree.

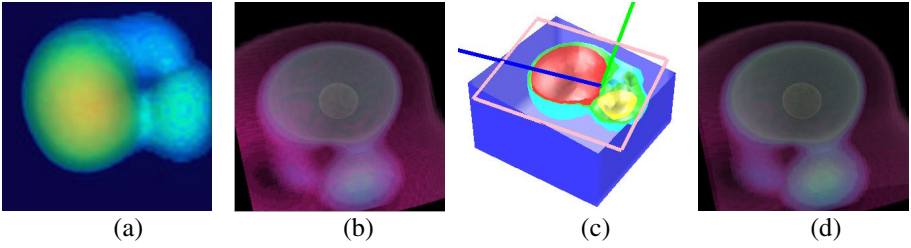


**Fig. 5.** Visualization of the National Library of Medicine (NLM) tooth volume: (a) Volume-rendered image of target datasets using an accentuated transfer function; (b) Volume-rendered image of cross-section; (c) Cutting off a polygon; (d) Volume-rendered image of the dataset on the far side of the cutting plane.

Figure 5(a) shows a volume-rendered image of the target dataset using an accentuated transfer function; Figures 5(b), (c), and (d) show cross-sections of the dataset. Figure 5(b) shows a volume-rendered image of the cross-section; Figure 5(c) shows a cross-section obtained by cutting off a polygon; and Figure 5(d) shows a volume-rendered image of the dataset on the far side of the cutting plane.

## 4.2 Antiproton-Hydrogen Atom Collision

The second example is an antiproton-hydrogen atom collision at an intermediate-collision energy in which a single antiproton collide with a single hydrogen atom. Details of the formulation and established numerical schemes can be found in [22].



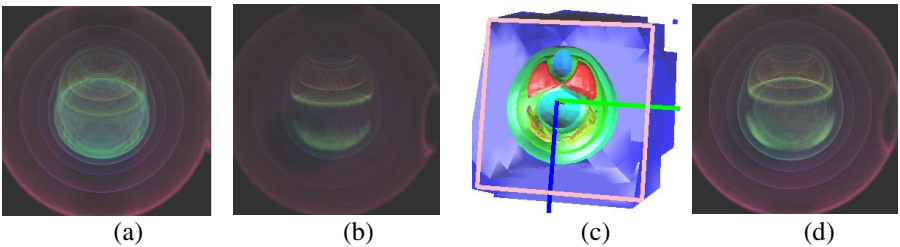
**Fig. 6.** Visualization of an antiproton-hydrogen atom collision: (a) Volume-rendered image of the target datasets using an accentuated transfer function; (b) Volume-rendered image of the cross-section; (c) Cutting off a polygon; (d) Volume-rendered image of the dataset on the far side of the cutting plane.

Figure 6(a) shows a volume-rendered image of the target dataset using an accentuated transfer function; Figures 6(b), (c), and (d) show cross-sections of the dataset. Figure 6(b) shows a volume-rendered image of the cross-section; Figure 6(c) shows a cross-section obtained by cutting off a polygon; and Figure 6(d) shows a volume-rendered image of the dataset on the far side of the cutting plane. By using VST, the system successfully finds the characteristic cross-section that contains a point where the proton collides with the hydrogen atom.

### 4.3 Nucleon

The last target is a ‘nucleon’ dataset, obtained by simulating the two-body distribution probability of a nucleon in the atomic nucleus of  $^{16}\text{O}$  [23]; its resolution here is  $41 \times 41 \times 41$ .

Figure 7(a) shows a volume-rendered image of the target datasets using an accentuated transfer function; Figures 7(b), (c), and (d) show cross-sections of the dataset. Figure 7(b) shows a volume-rendered image of the cross-section; Figure 7(c) shows a cross-section obtained by cutting off a polygon; and Figure 7(d) shows a volume-rendered image of the dataset on the far side of the cutting plane. Figure 7(c) demonstrates that a cross-section actually passes through all the critical isosurfaces.



**Fig. 7.** Visualization of the nucleon: (a) Volume-rendered image of target datasets using an accentuated transfer function; (b) Volume-rendered image of the cross-section; (c) Cutting off a polygon; (d) Volume-rendered image of the dataset on the far side of the cutting plane.

## 5 Conclusion and Future Work

This paper proposed a method for automatically generating characteristic cross-sections a given volume dataset of based on topological structure. Furthermore, we have implemented a prototype system to prove the effectiveness of the method and performed an experiment with various datasets. Existing system requires the user's knowledge and experience to find characteristic cross-sections. We automated the process and confirmed that our system can find appropriate cross-sections without tedious manual control.

In the future, we plan to extend our algorithm to generate cross-sections including curved surfaces and multiple planes. We also intend to try other methods, such as medial axes and generalized symmetry for analyzing inner structures, to apply automatic cross-sectioning to other model representations, in particular surface models. Finally, we would like to design better user interfaces for examining volume datasets.

## References

1. He, T., Hong, L., Kaufman, A., and Pfister, H.: Generation of Transfer Functions with Stochastic Search Techniques. *Proceedings of IEEE Visualization '96*, (1996), 227-234.
2. Marks, J. *et al.*: Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation. *Computer Graphics (Proceedings of Siggraph 97)*, (1997), 389-400.
3. Castro, S., König, A., Löffelmann, H., and Gröller, E.: Transfer Function Specification for the Visualization of Medical Data. *Technical Report TR-186-2-98-12*, Vienna University of Technology, (1998).
4. Fang, S., Biddlecome, T., and Tuceryan, M.: Image-Based Transfer Function Design for Data Exploration. *Proceedings of IEEE Visualization '98*, (1998), 319-326.
5. Kindlman, G., and Durkin, J.W.: Semi-Automatic Generation of Transfer Functions for Direct Volume Rendering. *Proceedings of 1998 IEEE Symposium on Volume Visualization*, (1998), 79-86.
6. Fujishiro, I., Azuma, T., and Takeshima, Y.: Automating Transfer Function Design for Comprehensible Volume Rendering Based on 3D Field Topology Analysis. *Proceedings of IEEE Visualization 1999*, (1999), 467-470.
7. Lyness, C., and Blake, E.: Real Time Isosurface Browsing. *Proceedings of the 1st International Conference on Computer Graphics, Virtual Reality and Visualization*, (2001), 143-146.
8. Itoh, T., and Koyamada, K.: Automatic Isosurface Propagation Using an Extrema Graph and Sorted Boundary Cell Lists. *IEEE Transactions on Visualization and Computer Graphics*, **1**(4), (1995), 319-327.
9. Owada, S., Nielsen, F., Okabe, M., and Igarashi, T.: Volumetric Illustration: Designing 3D Models with Internal Textures. *ACM Transactions on Graphics*, **23**(3), (*Proceedings of Siggraph 2004*), (2004), 322-328.
10. Hinckley, K., Pausch, R., Proffitt, D., and Kassell, N.: Two-Handed Virtual Manipulation. *ACM Transactions on Computer-Human Interaction (TOCHI)*, **5**(3), (1998), 260-302.
11. Viega, J., Conway, M. J., Williams, G., and Pausch, R.: 3D Magic Lenses. *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology*, (1996), 51-58.

12. Diepstraten, J., Weiskopf, D., and Ertl, T.: Interactive Cutaway Illustrations. *Computer Graphics Forum*, **25**(4), (*Proceedings of Eurographics 2003*), (2003), 523-532.
13. Takahashi, S., Takeshima, Y., and Fujishiro, I.: Topological Volume Skeletonization and its Application to Transfer Function Design. *Graphical Models*, **66**(1), (2004), 22-49.
14. Takahashi, S., Fujishiro, I., and Takeshima, Y.: Interval Volume Decomposer: A Topological Approach to Volume Traversal. to Appear in *Proceedings of SPIE Conference on Visualization and Data Analysis 2005*, (2005).
15. Itoh, T., Yamaguchi, Y., and Koyamada, K.: Fast Isosurface Generation Using the Volume Thinning Algorithm. *IEEE Transactions on Visualization and Computer Graphics*, **7**(1), (2001), 32-46.
16. Bajaj, C.L., Pascucci, V., and Schikore, D.R.: The Contour Spectrum. *Proceedings of IEEE Visualization '97*, (1997), 167-173.
17. Carr, H., Snoeyink, J., and Axen, U.: Computing Contour Trees in All Dimensions. *Computational Geometry*, **24**(2), (2003), 75-94.
18. Chakravarty, I. and Freeman, H.: Characteristic Views as a Basis for Three-Dimensional Object Recognition. *Proceedings of SPIE Conference on Robot Vision*, (1982), 37-45.
19. Agrawala, M., Phan, D., Heiser, J., Haymaker, J., Klingner, J., Hanrahan, P., and Tversky, B.: Designing Effective Step-By-Step Assemble Instructions. *ACM Transactions on Graphics*, **22**(3), (*Proceedings of Siggraph 2003*), (2003), 828-837.
20. Pfister, H., Lorensen, B., Bajaj, C., Kindlmann, G., Schroeder, W., Avila, L. S., Martin, K., Machiraju, R., and Lee, J.: The Transfer Function Bake-off. *Computer Graphics and Applications*, **21**(3), (2001), 16-22.
21. Woo, T.: The National Library of Medicine of the National Institute of Health [<http://visual.nlm.nih.gov/data/>].
22. Suzuki, R., Sato, H., and Kimura, M.: Antiproton-Hydrogen Atom Collision at Intermediate Energy. *IEEE Computing in Science and Engineering*, **4**(6), (2002), 24-33.
23. Meibner, M.: Web Page [<http://www.volvis.org/>].

# Interface Currents: Supporting Fluent Collaboration on Tabletop Displays

Uta Hinrichs<sup>1</sup>, Sheelagh Carpendale<sup>2</sup>, Stacey D. Scott<sup>2</sup>, and Eric Pattison<sup>2</sup>

<sup>1</sup> Department of Computer Science, Otto-von-Guericke University,  
Magdeburg, Germany

`uta.hinrichs@student.uni-magdeburg.de`

<sup>2</sup> Department of Computer Science, University of Calgary,  
Calgary, Alberta, Canada  
{sheelagh, sdscott, ericp}@cpsc.ucalgary.ca

**Abstract.** Large horizontal displays provide new opportunities to support individual and collaborative activities such as creativity and organizational tasks. We present Interface Currents, a fluid interaction technique designed to support face-to-face collaboration by improving access to and sharing of workspace items. Interface Currents are flexible containers that provide a controllable flow of interface items that support creativity during collaborative tasks and enable intuitive organization and sharing of digital information around horizontal displays.

## 1 Introduction

Many types of digital endeavours would benefit from more working space than is offered by common desktop displays. A single user trying to work with a large volume of materials or a group of users performing a planning, organizing, or brainstorming activity may benefit from a larger digital workspace. High-resolution, large-screen tabletop displays offer great potential for enabling such activities. Interactive tabletop displays have been developed to support activities such as information browsing [8,12,13,14,15], photo sharing [12], and document sharing [16]. However, with the advantages of increased size, these large displays come hand in hand with new interfaces issues.

Large tabletop displays afford quite different interactions than common 21 inch desktop displays and, thus, introduce several issues for interface designers that vary considerably from those investigated in Xerox Star [6] derivative desktop interfaces. For example, workspace items may be out of reach because they are located on the opposite side of the table from a user's current position. Furthermore, reaching across the table can also have territoriality issues during collaboration [10]. For instance, it is socially awkward to reach across another person's work or into that person's workspace [10]. Thus, interfaces for tabletop displays must provide better access to workspace items for both individual and collaborative interactions. They should also support easy passing and sharing

of workspace content among multiple users. People may also move around the table or adjust their seating arrangements as others join and leave the group and to suit their task needs [1,11]. Thus, large-display interfaces need to support mobility at the display, and therefore should not be designed to assume fixed user locations.

Interfaces for tabletop displays need to provide better access to workspace items whether they are in easy or hard to reach regions for both individual and collaborative interactions. They should provide greater ease for spreading, passing and sharing of workspace content and they should address the need for personal mobility during the creative process. To address these problems, we introduce the Interface Current, an interface component that can support a fluid and flexible interaction between users. Currents can facilitate access to workspace items and more readily support creative activities for either individuals or groups.

## 2 Related Work

While the concept of an Interface Current has not previously been considered as a type of interface component, several existing interfaces can be discussed in terms of a current. Other mechanisms have a facility for sharing and passing on large displays or incorporating fluidity into their interfaces.

The Interactive Table [8] at the Museum of Modern Art used a physical Lazy Susan on a table to display information, similar in concept to the Lazy Susan shown in Figure 1(a). Sharing and discussing the information between the visitors was stimulated and simplified by the Lazy Susan. The Interactive Table shows a way to present information intuitively to multiple users. However, in that system, the Lazy Susan is a physical device. In contrast, the Personal Digital Historian (PDH) [2,12] tabletop interface provides a digital, circular workspace that can be rotated like a virtual Lazy Susan. Applying an Interface Current to a region-based interface component provides an easy mechanism for creating a virtual Lazy Susan. In contrast to the virtual Lazy Susan on the PDH, Interface Currents support flexible, adjustable boundaries.

The Café Table [15] has a “Living Memory Flow Zone” around the table edge. In this zone, information objects flow. The “Flow Zone” was designed to support opportunistic browsing, which allows users to watch objects as they move by and to select desired objects either by stopping the flow with a touch or removing the chosen objects. However, the “Flow Zone” on the Café Table is not repositionable nor are its boundaries adjustable, features that are both provided by Interface Currents.

Ståhl and Lundberg’s Pond [14] tabletop interface presents data elements in a three-dimensional virtual pond, providing a new interaction metaphor for searching and browsing digital media. Media items related to a recent search query float to the surface of the Pond. Items which have not been interacted with for some time sink to the bottom of the Pond, gradually disappearing. This garbage collection also happens if the Pond becomes crowded. While the Pond does not allow users to directly control the properties of the floating behaviour,

this behaviour is similar to an Interface Current in that it uses the notion of a flow direction (up and down) and a flow speed. Geißler [4] introduces a gesture technique called "throwing" for large displays that supports moving items across large distances in a natural way. However, studies have shown that this technique is too imprecise [1]. Baudisch et al. [1] developed the "Drag-and-Pick" technique that augments the "Drag-and-Drop" technique for large displays. When a user drags an item, icons that are of a compatible type and located in the direction of the users' drag, produce copies of themselves called tip icons. These tip icons are connected to the source icon with a visual rubber band and move towards the dragged item to facilitate reach. On a shared, collaborative workspace, it could be confusing to have several crossing rubber bands invoked by multiple users.

Continuous series of images such as digital video data are often referred to as "streams". Several techniques have emerged that honour the fluid spatial/temporal quality of this data. One is to provide "fluid interaction" by creating a trail of video frames and by introducing some flexibility through the use of lens technology [9]. Another approach to creating interaction for video data is to present the trail in 3D, providing context for browsing [18].

### 3 The Concept of an Interface Current

Problems related to reach, mobility, and sharing also exist in traditional work and social environments. Physical devices have been developed to mitigate these problems. Consider the various "Lazy Susan"-type devices used in homes and restaurants for sharing and accessing food and other items on tabletops (Figure 1(a)), the water canals and conveyer belts used in Sushi restaurants to distribute food to customers seated at various locations at the Sushi bar (Figure 1(b)), rotating shelves in kitchen cupboards to access items stored in the cupboard (Figure 1(c)), and the airport luggage carousels (Figure 1(d)). These devices facilitate handling large amounts of items, and reducing the effort involved in accessing hard to reach items. The tabletop Lazy Susan (Figure 1(a)) also eases sharing items with others at a table and allows people to be located at various positions around the table. The concept of Interface Currents draws



**Fig. 1.** Traditional Lazy Susans: (a) wooden tabletop Lazy Susan, (b) Sushi conveyer belt, (c) cupboard Lazy Susan, (d) luggage conveyer belt.

on the physical solutions (Figure 1), as well as existing interface approaches, to provide a solution with additional flexibility.

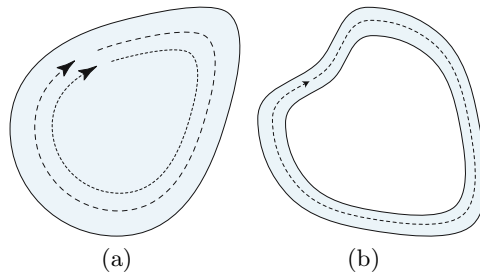
A current is a continuous onward movement, traditionally thought of as existing in a body of liquid or gas. Characteristically, an Interface Current will move or run smoothly with unbroken continuity in a manner similar to a fluid. It has directional flow, speed, containment boundaries, and a location. Workspace items such as images, documents, or tools can be placed on a Current and be affected by its flow.

### 3.1 Properties of an Interface Current

The fundamental properties of an Interface Current are *flow* and the *path* in which this flow travels. Depending on usage, the Interface Current's visibility can also be changed.

**Flow:** Integral to the idea of a Current is its ability to flow. The flow has a specific direction and speed, which can be adjusted separately or in combination. In our prototype, this flow is invisible unless an item is placed on the Current.

**Path:** A Current flows in a path that has a location and some type of boundaries. The location of the path can either be fixed or mobile. The Current's boundaries define its shape and size, indicating areas in the workspace that are affected by its flow and areas that are not. In the extreme case the entire interface can be affected by a Current and the boundaries can be the edges of the workspace. Boundaries can be rigid in that they are established initially and not changed, or they can be flexible and controlled either by the system or by the user. A single external boundary creates a pool (Figure 2(a)) and an additional internal boundary makes a stream (Figure 2(b)).



**Fig. 2.** Examples of Interface Currents: (a) pool with a Current and (b) stream

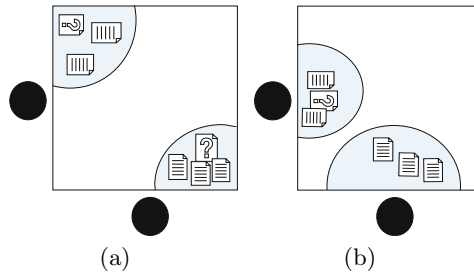
**Visibility:** While the path of an Interface Current determines its visual appearance, the location of the path affects its visibility. Depending on the function of the Current, its path can be totally visible, partly visible, or invisible. If an Interface Current is used as a personal storage area it can be reasonable to move it partly out of the workspace in order to save personal space (Figure 3).



In contrast, Currents used in a group space for storing the results of a group brainstorming session, can be totally visible (e.g., the central Current shown in Figure 11(a)). Any workspace items that are not currently needed can be moved out of the workspace (and retrieved when needed) by simply placing them on an Interface Current and moving the part or most of the Current out of the workspace.

### 3.2 Purposes of an Interface Current

**Providing a Group Space:** A group space is a shared work area on a tabletop display. Interface Currents in a group space can be used for organizing items that are accessible for all users at the display. Example group space tasks include brainstorming or design.



**Fig. 3.** Partly visible Interface Currents create more personal space

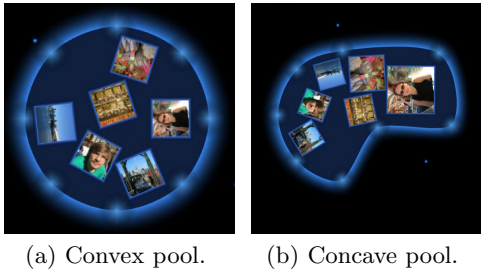
**Providing Personal Storage Areas:** Small or partly visible Interface Currents can be used for storing personal items. Partly visible Currents offer the opportunity to store large amounts of items, without crowding the personal workspace (Figure 3). The flow on the Current facilitates scrolling through the items in order to make invisible items visible and vice versa.

**Facilitating Item Sharing:** Interface Currents can be stretched out over a tabletop display in order to facilitate passing items. For instance, a Current can be used along the periphery of the tabletop workspace for passing content from one side of the table to the other (Figures 5(c) & 11), similar to the sushi conveyor belt in Figure 1(b). A user can adjust the velocity of the flow of items to control the rate of item passing.

### 3.3 Examples of Interface Currents

To manifest a Current in an interface, decisions must be made about the Current's flow, path, boundaries, location, presentation, and the relationship it has to other inter-face components. In order to explore these issues, we have implemented two main types of Current-based interface components: Pools and Streams. User experiences with these two types of components have shown that each appears to support different types of interactions and task activities.

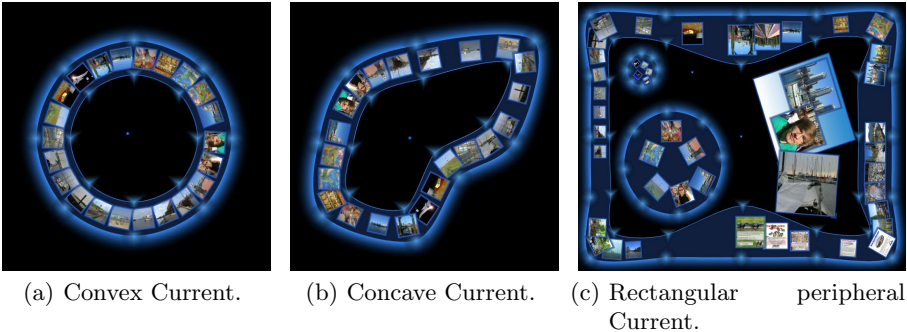
**Pools:** The concept of a pool-shaped Current, or a Pool, comes most directly from the Lazy Susan-type devices (Figures 1(a) & 1(c)). Rotating an entire workspace is an action common with traditional tabletop collaboration [12] and one that several existing tabletop interfaces have implemented [8,12,13,17]. Pools are bordered by only one single exterior boundary. Our implementation of a Pool shows the area as a translucent colour and the boundary as an attenuated line, with several small control points along the boundary (Figure 4). Our default Pool boundary is a circle (Figure 4(a)). However, its boundary or circumference can be adjusted to almost any shape (e.g., Figure 4(b)). A Pool can be expanded



**Fig. 4.** Adjusting the boundary of a Pool

or shrunk; a user can pull a region towards themselves or push it out of their way if it interferes with their activities. The boundary control points can be used to adjust the boundary shape through the use of a Flow-type menu [5], invoked by touching the control point. The only indication of the Current’s motion is the effect it has on the items placed on the Current. In order to solve the orientation problem on a tabletop display [7] all items are oriented towards the Pool’s boundary. Once the flow on a Pool is started, items follow the shape of its boundary.

**Streams:** The concept of a stream-shaped Current, or a Stream, comes from streams of water. They are similar to the little “stream” in a sushi restaurant that carries various sushi selections to customers and to the airport luggage

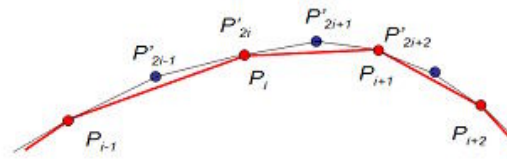


**Fig. 5.** Adjustable stream-shaped Currents

conveyor belts (Figures 1(b) & 1(d)). Our implementation shows a Stream as a ribbon of translucent color with an attenuated line along its outer boundary, a thin line along its inner boundary and small control points along each boundary (Figure 5). Streams can also be adjusted through a Flow-type menu [5] invoked via the boundary control points. They are adjustable to nearly any desired shape (e.g., Figure 5(b)). Items on Streams move parallel to its boundaries. Similar to items contained in a Pool, they are automatically oriented its outer boundary.

### 3.4 Interacting with Interface Currents

**Controlling the Appearance of the Current:** To create a flexible, dynamically adjustable boundary we have used a basic four-point interpolating subdivision curve [3], as illustrated in Figure 6. All points generated by this method are on the boundary. We find the C1 continuity acceptable given that this method provides direct interactive control. We use eight (for peripheral Currents sixteen) initial points that are always visible in the interface, and five subdivision levels. Touching any of these points either by finger or by stylus, invokes a Flow-type menu [5] menu that offers different types of control over the Current's shape (e.g., Figures 7(a), 7(b) & 7(c)).

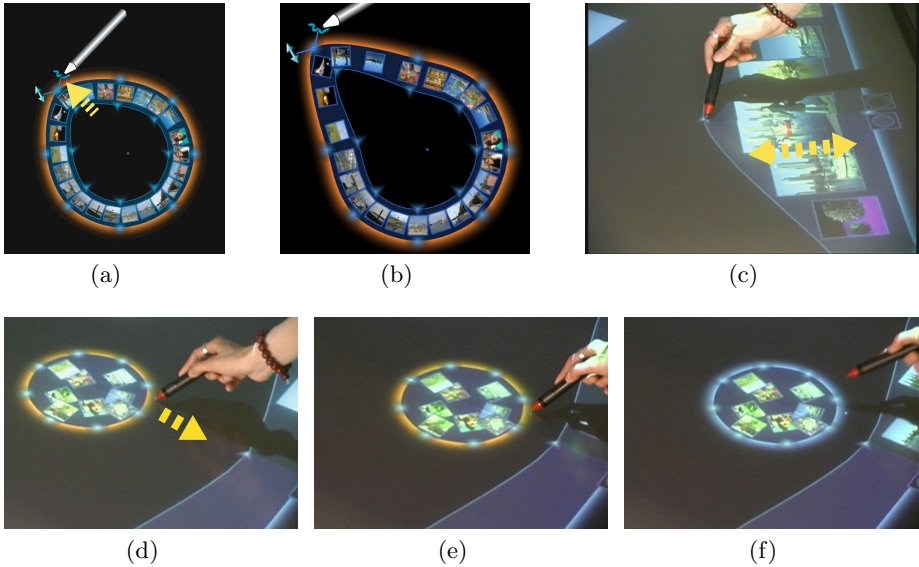


**Fig. 6.** Diagram of the 4-point interpolating subdivision algorithm

**Adjusting the Boundary:** Moving one's finger or a stylus while touching a menu icon in the Flow-type menu will change the geometric shape of the Interface Current. If it is a Pool, only the outer boundary changes: if it is a Stream, the path followed by both inner and outer boundaries changes (Figures 7(a) & 7(b)). With a Stream, however, it is also possible to manipulate the inner boundary alone in order to change the width of Stream at that location (Figure 7(c)). Note that items on the Stream magnify or shrink depending on the Stream's width at that location. Thus, width of the Stream does not necessarily have to be constant along its entire flow path (Figure 7(c)). In general, boundary adjustments allow any amount of stretching or compression of the Interface Current's size, no matter to which type of Current it is applied.

**Manipulating the Currents' Position:** An Interface Current can be moved to any location in the workspace, by simply touching and dragging on its outer boundary (Figures 7(d), 7(e) & 7(f)). It can even be positioned partially outside the workspace and, thus, becoming partly invisible. Since easy access to the

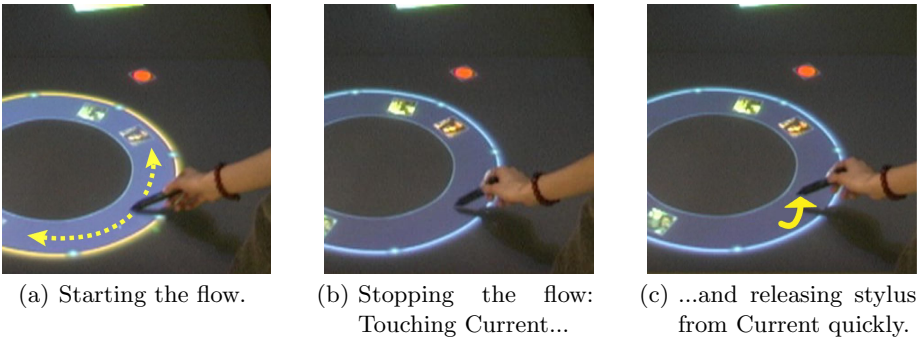
Current's flow still makes all items contained with the Current readily available, this expands useable space in a manner similar to use of Lazy Susans in kitchen cupboards (Figure 1(c)). As shown in Figure 5(c) it is possible to have multiple Interface Currents in the workspace. Through repositioning overlapping of different Currents can happen. In this case, floating items stay with the Currents to which they belong. There is no confusion of items moving from one Current to the next, unless that item is explicitly put there by a person.



**Fig. 7.** Interaction with Currents: (a + b) adjusting a stream's shape both inner and outer boundary; (c) adjusting the inner boundary only; (d-f): relocating a Current.

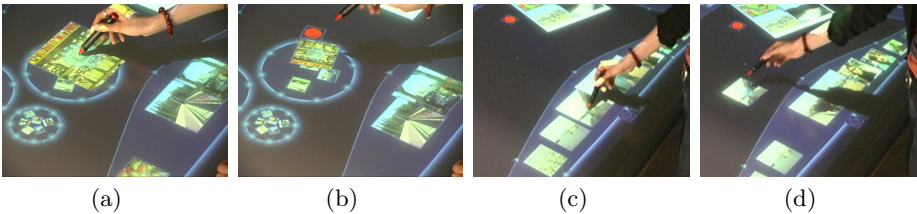
**Controlling the Flow on an Interface Current:** By default, an Interface Current is floating continuously. In order to change the direction of the flow, a user simply has to indicate a new direction by touching inside the Current's boundaries (Figure 8(a)). Thus, if the items on the current are floating clockwise, moving the touch input counter-clockwise will reverse the direction of the flow. A brief touch on the Current will stop the flow and another contact with motion will start it again (Figures 8(b) & 8(c)). The length of the gesture used to start or change the flow determines the speed of the flow. Our current implementation enables users to adjust the flow to three different speeds. The appropriate speed of the flow may depend on the task being performed.

**Current-Item Interaction:** An Interface Current does not need to contain any interface items. It can be simply an area in the workspace in which items can be placed for accessing, passing, and sharing. Workspace content, such as



**Fig. 8.** Manipulating a Current's flow

images and text items, can be placed in it or removed from it. Figures 9(a) & 9(b) show the addition of an image to a Pool. As soon as the image is placed on the Current, it is automatically resized to fit the size of the Current at that location. Content that is removed from a Current (see Figures 9(c) & 9(d)) is returned to its original size.



**Fig. 9.** Placing an item on a pool-based Current: (a) approaching the Current, (b) releasing the item onto the Current; (c-d) Removing an item from a Stream: (c) selecting the item in the Stream, (d) leaving the Stream.

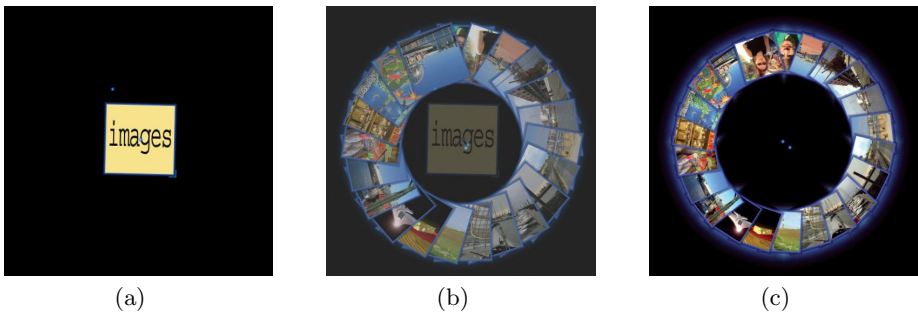
**Transforming Folders into Currents:** To share content of folders easily between several people working on a horizontal display, folders can be transformed into Interface Currents. Double-clicking a folder in the workspace will evoke a folder-to-Current transformation (see Figures 10(a) & 10(b)). This animated transformation results in a Stream-Current that includes the content of the folder spread out evenly and flow smoothly (see Figure 10(c)). The reverse transition can be activated changing a folder-generated Current back to a folder again. This can be very useful to save space in the workspace. The interaction with a Current created from a folder is the same as other Currents: The contents of a folder-Current can be adjusted and items can be added and removed. Adding and removing items will affect the contents of the associated folder.

When traditional folders are opened, their contents are arranged according to a single alignment. On a tabletop display, this single alignment of contents can

make them difficult to view and interact with for those group members located at different sides of the table. Opening the folder to a Current can orient items towards the Current's outer edge, which is more appropriate when there are group members all around the table. Users could browse through the contents easily by activating the Current's flow.

## 4 Discussion

Interface Currents facilitate information sharing and access on tabletop displays. Through the combination of motion on demand and the high degree of flexibility, our Interface Current components support a large variety of tasks. Interface Currents can be used to ease access to items located at hard to reach areas of the workspace and provide a way of sharing information between people. For example, a Current can be stretched around the perimeter of a tabletop display so that a group of people can share access to a set of information (Figure 11) and be used for passing items back and forth. A Pool can be used to facilitate storing and sharing of items. Rotating a Pool using the Current's flow can bring far-away items closer, or it can be used to access workspace areas beyond the display space. Several Interface Currents can be used to spread information for brainstorming



**Fig. 10.** Opening a folder into a stream-shaped Current: (a) the collapsed folder, (b) transition stage (folder fades and the Current opens), and (c) expanded folder.

activities (Figure 11). Currents can be shrunk and used for personal information access. The control of the Current's flow and the flexibility of its path help facilitate sharing. The flexibility of the paths allows people to tailor the Currents for the task at hand. For instance, a group of people who are collaborating can set up a Stream to exchange information by arranging the Current so that it passes conveniently near each of them. Figure 11 shows users working on a table with a Stream stretched around the edge of the tabletop workspace to operate like a conveyer belt. When one of these collaborators temporarily needs to attend to a different task, the Current can simply be pushed away to prevent interference. Currents lend themselves to being tucked out of the way,





**Fig. 11.** A variety of uses of Interface Currents on a tabletop display

and by placing a considerable portion beyond the edge of the screen, extending the available screen space. A Current-based component may have two visible portions near opposite edges of the workspace and a connection between them that flows under the workspace. A user can place content on one of these visible portions and the current will carry the content under the workspace to appear on the other visible portion. This provides a transport or passing mechanism that does not interrupt work being performed in the main workspace area.

A user study exploring usage of and interaction with Interface Currents is underway. Preliminary results indicate that people with little computer experience found Interface Currents as supporting and intuitive to use, particularly for tasks that require creativity and the management of large amounts of information.

## 5 Conclusion

We have introduced the concept of Interface Currents to help address issues related to accessing and sharing of information at tabletop displays. A Current consists of a directional flow and a path in which this flow travels. We have presented several Current-based interface components, Pools and Streams which can be used to store, transport, and share workspace items. Current-based interface components such as Pools and Streams allow a user to manipulate workspace items indirectly, facilitating tasks such as reaching, passing, and sharing. Users can directly control the Current through easy mechanisms for starting, stopping, and changing the direction of the flow. We have designed these components with flexible boundaries that can be easily reshaped or repositioned to suit a users' immediate task needs. Currents introduce mobility into information access and organization. Mobility of workspace content is a step towards supporting the mobility of people using the interface. Workspace content can be readily moved as needed. Flexible boundaries support rapid and easy expansion, contraction, stretching and bending of regions that support information flow. Additionally, whole components such as Pools or Streams can be easily relocated. Interface Currents can be used to extend traditional interface components, such as folders, to improve the accessibility of these components on a tabletop display. Moreover,

Interface Currents offer many interesting opportunities for developing new types of interface components to assist individual and collaborative interactions.

**Acknowledgments.** We would like to thank Natural Sciences and Engineering Research Council of Canada, Alberta's Informatics Circle of Research Excellence, Alberta Ingenuity, and the Canadian Foundation of Innovation for research support. We also thank our fellow researchers from the Interactions Lab at the University of Calgary for their insightful comments on this work.

## References

1. Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P., Bederson, B., and Zierlinger, A.: Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch- and Pen-operated Systems. In *Proceedings of Interact'03*, 2003, pp. 57–64.
2. Dietz, P. and Leigh, D.: A Multi-User Touch Technology. In *Proceedings of UIST'01 Symposium on User Interface Software and Technology*, 4(4), 2001, pp. 219–226.
3. Dyn, N., Levin, D., and Gregory, J.: A Four-point Interpolatory Subdivision Scheme for Curve Design. *Computer Aided Geometric Design*, 4(4), 1987, pp. 257–268.
4. Geißler, J.: Shuffle, Throw or Take it! Working efficiently with an Interactive Wall. *CHI'98 conference summary on Human factors in computing systems*, ACM, 1998, pp. 265–266.
5. Guimbretière, F., Winograd, T.: Combining Command Text and Parameter Entry. In *Proceedings of UIST'00 Symposium on User Interface Software and Technology*, ACM: NY, 2000, pp. 213–216.
6. Johnson, J., Roberts, T. L., Verplank, W., Smith, D. C., Irby, C., Beard, M., Mackey, K.: The Xerox Star: A retrospective. *Computer*, 22(9), 1989, pp. 11–29.
7. Kruger, R., Carpendale, M.S.T., Scott, S.D., Greenberg, S.: Roles of Orientation in Tabletop Collaboration: Comprehension, Coordination and Communication. In *Journal of Computer Supported Collaborative Work*, 13(5-6), 2004, pp. 501–537.
8. Omojola, O., Post, E.R., Hancher, M.D., Maguire, Y., Schoner, B.: An Installation of Interactive Furniture. In *IBM Systems Journal*, 39 (3&4), 2000, pp. 861–878.
9. Ramos, G., Balakrishnan, R.: Fluid Interaction Techniques for the Control and Annotation of Digital Video. In *Proceedings of UIST'03 Symposium on User Interface Software and Technology*, ACM: NY, 2003, pp. 105–114.
10. Scott, S.D., Carpendale, M.S.T., Inkpen, K.M.: Territoriality in Collaborative Tabletop Workspaces. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW)'04*, 2004, pp. 294–303.
11. Scott, S.D., Grant, K.D., and Mandryk, R.L.: System Guidelines for Co-located, Collaborative Work on a Tabletop Display. In *Proceedings of ECSCW'03 Conference on European Computer-Supported Cooperative Work*, 2003, pp. 159–178.
12. Shen, C., Lesh, N., Moghaddam, B., Beardsley, P., and Bardsley, R.S.: Personal Digital Historian: User Interface Design. In *Extended Abstracts of CHI '01 Conference on Human Factors in Computing Systems*, ACM: NY, 2001, pp. 29–30.
13. Shen, C., Lesh, N., Vernier, F., Forlines, C., and Frost, J.: Building and Sharing Digital Group Histories. In *Proceedings of CSCW '02 Conference on Computer-Supported Cooperative Work*, ACM: NY, 2002, pp. 324–333.



14. Ståhl, O., Wallberg, A., Soederberg, J., and Humble, J.: Information Exploration Using the Pond. In *Proceedings of CVE'02 Conference on Collaborative Virtual Environments*, ACM: NY, 2002, pp. 72–79.
15. Stathis, K., de Bruijn, O., and Macedo, S.: Living Memory: Agent-Based Information Management for Connected Local Communities. *Interacting with Computers*, 14(6), 2002, pp. 663–688.
16. Streitz, N., Geißler, J., Holmer, T., Konomi, S., Müller-Tomfelde, C., Reischl, W., Rexroth, P., Seitz, P., and Steinmetz, R.: i-LAND: An Interactive Landscape for Creativity and Innovation. In *Proceedings of CHI '99 Conference on Human Factors in Computing Systems*, ACM: NY, 1999, pp. 120–127.
17. Vernier, F., Lesh, N., and Shen, C.: Visualization Techniques for Circular Tabletop Interfaces. In *Proceedings of AVI '02. Conference on Advanced Visual Interfaces*, ACM: NY, 2002, pp. 257–263.
18. Wittenburg, K., Forlines, C.: Rapid Serial Visual Presentation Techniques for Customer Digital Video Devices. In *Proceedings of UIST'03 Symposium on User Interface Software and Technology*, ACM: NY, 2003, pp. 115–124.

# Design of Affordances for Direct Manipulation of Digital Information in Ubiquitous Computing Scenarios

Lucia Terrenghi

University of Munich, Media Informatics,  
Amalienstrasse 17, 80333 Munich, Germany  
lucia.terrenghi@ifi.lmu.de

**Abstract.** This work focuses on interface design for supporting users of instrumented environments to interact with virtual information embedded in the real world. A gesture-based interaction paradigm is presented and the prototype of an interface providing affordances for gesture-based direct manipulation of digital information is described.

## 1 Introduction

The emergence of ubiquitous computing [18] scenarios promises to bring digital information on the walls of everyday life environments, embedding computing and displaying capabilities within the physical space and the material artifacts which populate it. In such a setting, where people can move around rather than sit at a desktop behind their screens, handle physical and digital objects at the same time, interact on shared displays in co-located or remote collaboration, the WIMP interaction paradigm comes short in supporting users' interaction. More natural ways of manipulating information need to be thought through. This paper points out the new challenges that mixed reality raises in terms of interaction design, and suggests a new gesture-based direct manipulation paradigm for instrumented environments, where technology is embedded in the physical space so as to display and sense information. The prototype of a user interface based on a mug metaphor developed within the FLUIDUM project [6] and introduced in [17] is described. In addition, a reflection on the new design perspective that needs to be taken when merging digital and physical information is presented.

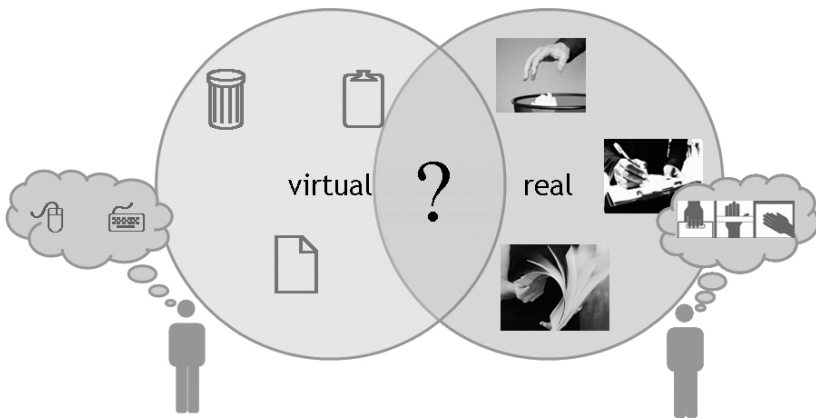
## 2 Direct Manipulation in Instrumented Environments

In the Personal Computer environment, direct manipulation describes the activity of manipulating objects and navigating through virtual spaces by exploiting users' knowledge of how they do this in the physical world [16]. The three main principles of direct manipulation are:

- continuous representation of the objects and actions of interest;
- physical actions or presses of labeled buttons instead of complex syntax;

- rapid incremental and reversible operations whose effect on the object of interest is immediately visible.

Direct manipulation is the basis for the dominant WIMP paradigm (Windows, Icons, Menu, Pointer), with which we manage different applications; according to the activities they support, applications rely on different metaphors. In the Microsoft Office software package, for instance, visual and auditory icons mimic the objects of a real physical office. In software programs for graphic design, icons resemble brushes and pencils. While the metaphor varies according to the domain (which translated to instrumented environments could be office, living room, kitchen, etc.), the general paradigm does not change. In addition, the appearance of widgets for desktop GUIs remains consistent, and suggests affordances for mouse and keyboard interaction, e.g. 3D effects for clicking buttons, white fields for text entry, ripples on the moving part of scrollbars for dragging. Although talking about direct manipulation, in the desktop environment we mostly need indirect input devices, such as mice, track pads or joysticks, to interact with the system. The desktop metaphor maps elements of the GUI to objects of the real world: those in turn provide affordances for gesture-based direct manipulation. When mixing virtual and real information the need of providing affordances for new interaction emerges (see Figure 1).



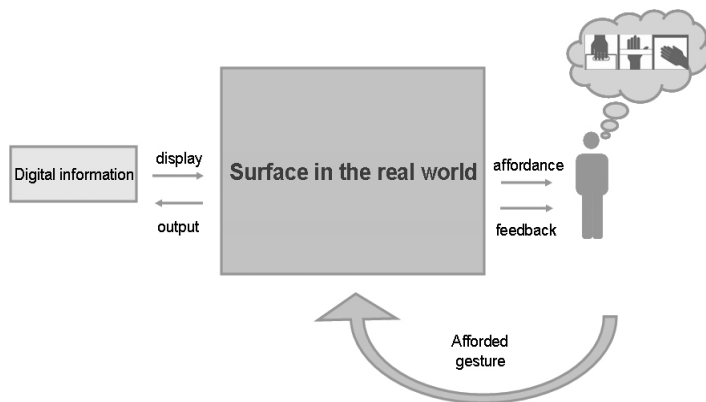
**Fig. 1.** In the Personal Computer environment classical GUIs rely on visual cues and metaphors in order to suggest the interaction operated with mouse and keyboard on virtual information. Real world objects provide affordances for manipulation. Affordances for information in mixed reality need to be designed for users' construction of an interaction conceptual model.

## 2.1 Related Work

With the emergence of ubiquitous computing scenarios, some work has been done to translate direct manipulation from the desktop environment. Butz et al. [1] try to transfer the window-desktop metaphor to 3D Augmented Reality Environments where each document type as well as each application has a visual representation in the form of a 3D icon. The operations on these icons are similar to the ones known from the

window-desktop metaphor, as for example drag and drop or click to open. Other research has been focusing on devices that can work as universal remote controllers [12] or support different functions depending on the way they are physically manipulated [13][2]. Other approaches look at natural input techniques such as gesture and gaze input [19], or to tangible user interfaces that work as tokens of information to be manipulated in the physical space [7].

In this paper we present a novel interaction paradigm, aiming at direct manipulation of units of information across different displays and contexts, avoiding the use of mouse and additional control devices. In such a paradigm, surfaces act as interfaces, and hands as control devices. Ringel et al. [15] have worked in a similar direction, looking at direct hands-on manipulation without implements on SMARTBoards: differently from such approach, our attempt is not to map mouse-based interaction to hands-based ones on a touch screen display. The mouse, indeed, has a limited manipulation vocabulary (e.g. click, double click, click and drag, right click) while hands, fingers and gestures provide a much more varied one (e.g. press, draw a circle, point, rotate, grasp, wipe, etc.). Rekimoto [14] exploits such variety working on a two-hands, multiple fingers gestures vocabulary. The limit of such work is that the user has to rely on the memory of a set of actions to be performed, in order to operate with the system: the memory of such set of actions is not supported by an explicit mapping between perception and action, which is the essence of affordances.



**Fig. 2.** The representation of abstract digital information should present affordances for gesture-based manipulation when appearing on the surface. In such a paradigm, surfaces play as interfaces and hands as control tools.

Our intent, therefore, is to design affordances for the representation of digital information which can suggest the hand gestures to be performed by the user (see Figure 2). Muscular memory can be used to structure frequent, patterned tasks into stylized or abbreviated gestures, thus providing the possibility to build a gesture vocabulary that exploits humans' sensorymotor perception [9] and haptic memory.

### 3 Design of an “Affordable” Interaction Metaphor

Objects of everyday life already carry information per se: their shape, color, texture, weight, temperature, material, all the aspects related to their physical features. Ecological approaches [5] focus on perception and action based on human attributes: in this context affordances can be described as properties of the world defined with respect to people’s interaction with it [4]. We perceive the environment directly in terms of its potential for action: in the physical experience we explore the outside world with our senses, making inferences of objects physical behavior thanks to a semantic of perception. This is the main source for the creation of users’ conceptual models of how things work. The concept of affordances is familiar especially to product design: Norman [10] applies the concept of affordances to every day life artifacts.

While designing affordances for digital information, two main design aspects can be addressed. On the one hand the visual appearance of the displayed information can suggest the gesture to be performed just relying on shapes and visual cues. Frames define semantic areas of interaction on the screen. 3D effects bring the information in fore- or back- ground and can suggest pressure. These visual cues can enhance perceptual and cognitive viewers’ processing of information [11]. When objects’ pliancy, i.e. their characteristic to be interactive [3], is hinted by the graphics, people get an idea of the action to be taken. In this case marks are, in semiotic terms, *nomically* related to their referents, rather than metaphorically, thus do not require viewers’ interpretation of symbols [4].

On the other hand the metaphoric link to real world objects and to their affordances in the physical world can provide rich material for the design of affordances for digital information. The virtual representation of a lever doorhandle, for instance, can be mapped to the natural gesture we do when we exercise some pressure on a real physical lever. The representation of a steering wheel can be mapped to the turning gesture we perform while driving. In this case the mapping relies on the analogy between the elements representing digital information, and the affordances of their referents in the physical space.

Metaphors have long been used in GUIs for providing an intuition of how things work using the world’s knowledge. While the desktop metaphor suits the type of environment in which the computing capabilities have been mostly applied so far, it runs short in scenarios of ubiquitous computing. Furthermore the visual affordances of the metaphoric items (e.g. folders and 2D icons) are suitable for the mouse-based manipulation vocabulary, but not for a hands-based one. Real world objects have affordances for manipulation and are embedded in conceptual models: digital representations of real world objects can rely on similar affordances and similar conceptual models. Building on these assumptions we have been working on the design of an interaction metaphor that

- is affordable for hands-based manipulation
- suits different environments.

## 4 Affordances Design Issues

In order to accomplish the direct manipulation principle of continuous representation of objects and actions, information needs to be consistently represented across a variety of contexts and to provide feedback. This helps to build a mental model: we form hypothesis of system function and physically probe and test them with our action. A preconceived model influences both the final impression and the manner in which we explore. Additionally, information may appear differently when associated with different objects, or when assuming a different status. In this sense the representation of virtual information should provide affordances that can be mapped to a certain status and suggest certain actions.

A main aspect of affordances is that physical attributes of the thing to be acted upon are compatible with those of the actor [4]: thus, representing digital information in an “affordable” way means to consider ergonomic aspects such as dominant hands, hands size, users’ height, and so on.

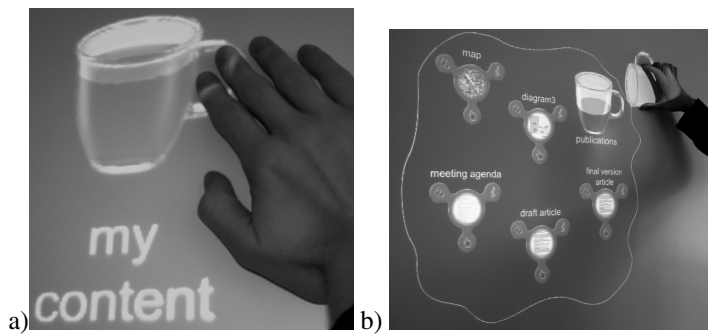
In the paradigm illustrated above, which describes surfaces as interfaces and hands as controls, the main differences between hands and mice as operating tools need to be taken into account. A first simple difference is that hands allow for multiple simultaneous inputs. Reflecting on how we manipulate physical objects, we can easily notice hands cooperative work. For instance, we usually hold a glass with the non-dominant hand and pour the content of a bottle with the dominant hand; we hold a vase with the non-dominant hand and open the lid by rotating it with the dominant one.

The fact that there is no spatial distance between physical input and digital output also implies additional considerations, such as shadows and visual angles. Furthermore, while the ratio between the pointer and the display sizes remain constant in a mouse-based interaction, i.e. the pointer area displayed on a screen scales proportionally to the screen size, in a hands-based interaction the ratio varies in function of hands sizes.

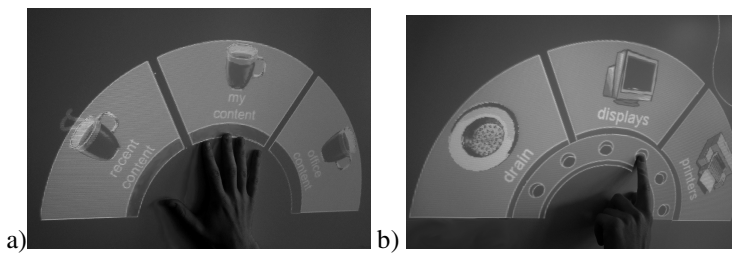
## 5 The Mug Metaphor Interface Prototype

The mug interface explores the idea to rely on the affordances provided by a mug, and to metaphorically represent it as a container of information. When manipulating a real mug we know we can move it around by holding its handle, and incline it to pour its content (Figures 3a, 3b). Empty mugs are expected to be lighter than full ones (e.g. contain less data), smoking mugs are expected to be hot (e.g. contain recent data). Additionally, a mug is an everyday life object which we use in different environments, e.g., in the office, in a living room, in a kitchen.

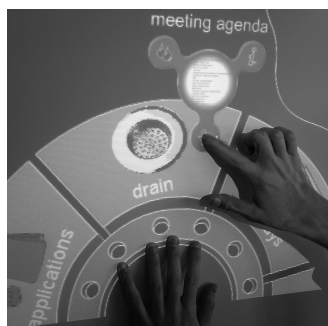
A first prototype of such a mug metaphor interface has been built in order to investigate the possibility to map the affordances of real world objects to gestures, relying on the conceptual model in which such real objects are embedded. In such a concept, mugs and units of information, the latter represented as kind of drops, can be



**Fig. 3.** The mug metaphor interface. a) To move the mug/information container, the user touches its handle and drags it on the screen surface. b) To explore its content the user turns the mug.



**Fig. 4.** a) The dominant hand is devoted to the management of information b) The non-dominant hand is devoted to management of resources.



**Fig. 5.** The two hands cooperate in manipulating information: the dominant one drags a unit of information on the drain to delete it

manipulated across the display. The dominant hand, e.g. the right one, is devoted to the manipulation and navigation of information. A pie menu displaying containers of information is displayed in correspondence of the right hand (Figure 4a). The non-dominant hand (e.g. the left one) works as command invocation, managing a menu of

resources (e.g. drain, displays, printers). Such menu can be scrolled with a movement of the finger on a holed gear, which makes the circle segments rotate (Figure 4b). The dominant hand moves units of information to the preferred resource (see Figure 5). The pie menus appear in correspondence of the hands, thus “following” the user while moving across the display, rather than being operable just in a fixed location on the screen. This responds to the need of freedom of movement of the user, and to enable two-hands cooperative interaction.

## 6 Conclusions and Discussion

In this paper an interaction paradigm for gesture-based direct manipulation of digital information in an instrumented environment was presented. In addition, the prototype of an interface based on a mug metaphor was introduced as example of ongoing research in the direction of affordances design for digital information.

### 6.1 Future Work

This work is still at an early stage and several issues concerning gesture recognition based on camera vision still need to be solved. Even though the paper focuses on the visual appearance of affordances, future work will explore auditory and haptic displays in order to provide redundancy and feedback. The tactile and audio channels, indeed, allow us to make sense of information in different ways. While focusing on a visual task, audio displays can influence our awareness of the environment: for instance, sound cues can draw our attention to different objects in the space (e.g. the noise of dripping water for pending tasks). Haptic displays can provide feedback to the action afforded by a visual cue: e.g., the force to be applied on a lever doorhandle can be coupled by a haptic feedback. In this sense the multimodality of interaction with digital information promises to provide more analogue interfaces.

### 6.2 A New Design Perspective

As emphasized in this paper, the design of affordances for digital information embedded in a real physical environment implies the consideration of new aspects which differ from the desktop PC environment. The users’ possibility to move around in the space and to directly manipulate objects and information items needs to be supported by interfaces that are properly scaled to users’ metrics, locations in the space, motor capabilities. The physicality of the real environment in which interaction takes place will have an impact on the interaction itself: issues such as the height of the user, her visual angle, the reachability of displayed objects to the hands, the proportion between objects and hands sizes, are some examples. In order to face such issues, ergonomic considerations need to be included in the interface design, thus suggesting the emergence of a novel design practice. The traditional usability guidelines for visual displays [8] will most likely need to be revised in order to address the novel aspects brought by ubiquitous computing. In these scenarios we expect that the design discipline will need to merge screen and product design competences, in order to merge virtual and physical worlds.



**Acknowledgments.** This research has been funded by Deutsche Forschungsgemeinschaft (DFG) within the FLUIDUM (Flexible User Interfaces for Distributed Ubiquitous Machinery) project.

## References

1. Butz, A., Beshers, C., Feiner, S., "Of Vampire Mirrors and Privacy Lamps: Privacy Management in Multi-User Augmented Environments", ACM UIST Symposium on User Interface Software and Technology, San Francisco, 1998.
2. Cao, X., Balakrishnan, R., VisionWand: Interaction Techniques for Large Displays using a Passive Wand Tracked in 3D. ACM UIST Symposium on User Interface Software and Technology, 2003.
3. Cooper, A., Reimann, R., About Face 2.0: The Essentials of Interaction Design. Wiley, 17 March, 2003.
4. Gaver, W., Technology Affordances. In Proc. ACM CHI 1991.
5. Gibson, J. J., The Ecological Approach to Visual Perception. Houghton Mifflin, New York
6. FLUIDUM project, FLEXible User Interfaces for Distributed Ubiquitous Machinery, <http://www.fluidum.org>
7. Ishii, H., Ullmer, B. Tangible Bits: towards seamless Interfaces between People, Bits, and Atoms. In Proc. CHI 1997.
8. ISO 9241-11:1998. (1998). Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability.
9. O'Regan and A. Noe, "On the Brain-basis of Visual Consciousness: A Sensory-Motor Approach", in Vision and Mind, ed. A. Noe and E. Thompson, MIT Press, 2002.
10. Norman, D.A., The Psychology of Everyday Things. Basic Books, New York, 1998.
11. Paley, W.B., Illustrative Interfaces: Building Special-Purpose Interfaces with Art Techniques and Brain Science Findings. In Proc. Smart Graphics 2003, Heidelberg, Germany.
12. Rekimoto, J. Pick and Drop: a Direct Manipulation Technique for Multiple Computer Environments. In Proc. UIST'97, (1997).
13. Rekimoto, J., Sciammarella, E.: ToolStone: Effective Use of the Physical Manipulation Vocabularies of Input Devices. ACM UIST Symposium on User Interface Software and Technology. p. 109-117, 2000.
14. Rekimoto, J. SmartSkin: an Infrastructure for Freehand Manipulation in Interactive Surfaces, ACM Press, CHI 2002, 113-120.
15. Ringel, M., Berg, H., Jin, Y., Winograd, T. Barehands: Implement-Free Interaction with a Wall-Mounted Display. ACM CHI Conference on Human Factors in Computing Systems (Extended Abstracts) p.367-368., 2001.
16. Shneiderman, B., Direct manipulation: A step beyond programming languages, IEEE Computer 16, 8, August 1983, 57-69.
17. Terrenghi, L.: Design for Interaction in Instrumented Environments. To appear in Doctoral colloquium proceedings of Pervasive 2005, May 2005, Munich, Germany.
18. Weiser, M.: The computer for the 21st century. Scientific American, Vol. 265, September 1991.
19. Zhai, S., Morimoto, C. Ihde, S.: Manual and Gaze Input Cascades (MAGIC) Pointing. In Proc. CHI'99.

# 2D Drawing System with Seamless Mode Transition

Yoshihito Ohki and Yasushi Yamaguchi

University of Tokyo,  
3-8-1 Komaba Meguro-ku,  
Tokyo, Japan  
+81-3-5454-6832  
{yohki, yama}@graco.c.u-tokyo.ac.jp

**Abstract.** Indirect and unintuitive commands like mode-switching are difficult operations for many users. In this paper, we describe a new simple interface for a 2D drawing system to lessen user's confusion. Firstly, we propose a direct manipulation interface based on a set of geometric primitives and their geometric relationships. Secondly, we present a seamless drawing interface that users do not need to switch modes by clicking the toolbar. Through user tests using our prototypes and a standard commercial application, we confirmed that our system can complete certain drawing tasks with shorter completion time and less number of commands.

## 1 Introduction

In many drawing systems, selecting appropriate tools or modes in a right order to complete a task is difficult and time-consuming, especially for non-expert users. As graphic systems support more complicated functionalities, more tools and modes are introduced, which lead to more difficulty. In this paper, we describe a new 2D drawing system which is designed to lessen the burdensome operations, e.g. mode-switching and command execution driven by menu selection, and to provide much simpler user interface.

### 1.1 Target Users and Shapes

Our system focuses on users who are not familiar with graphics design, but still want to draw some 2D shapes like simple figures in a business documents, or icons and symbols for personal web pages. Shapes shown in Fig.1 are some examples drawn by our system. All shapes must be bounded by a closed loop. In other words, any shapes with open segments are not supported by this system. Notice that these shapes are composed of simple geometric primitives, and these primitives satisfy certain geometric constraints such as congruency, symmetry or parallelism. Our system mainly handles these geometric primitives and their geometric constraints.



**Fig. 1.** Example shapes drawn by our system

## 1.2 Related Work

There are some researches proposing automatic inference framework to solve geometric constraints among existing objects, to avoid execution of an indirect command by users. In Snap-Dragging[1], the system automatically generates alignment objects to which the mouse pointer is snapped. HyperSnapping[5] infers geometric constraints such as grouping or repetition from user's aligning operations. Briar[2] also handles the geometric constraints. It allows a user to tell their intentions by specifying additional constraints.

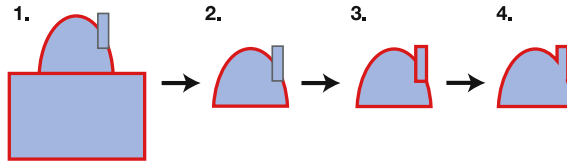
Raisamo[6] presented a stick metaphor which enables direct manipulation on drawing objects such as alignment, carving, cutting and rotating. It relieves a user of menu selection for executing a command which is not a direct and intuitive action. Honda[3] proposed an integrated way of manipulating drawing objects based on starting point's relative location to the target object. The system provides direct manipulation as well as modeless interaction. Pegasus[4] adopted a stroke-based interface using a pen device which also enables direct manipulation and modeless interaction. It is also capable of maintaining general geometric constraints such as parallelism, orthogonality, even intervals, and so on. However, it is difficult for freehand input system to avoid input errors and to maintain accuracy.

Our interface for handling primitives is partly similar to Raisamo's stick metaphor[6], but switching modes is much simpler in our system. An alternative way to avoid mode-switching is to use freehand recognition. Pegasus[4] is a free-hand beautification system based on constraint solver. We do not adopt freehand input since it inherently contains input errors and is inferior in accuracy.

## 2 Direct Manipulation of Geometric Primitives

### 2.1 Basic Idea and Framework

In this system, users create a shape by adding or subtracting primitives, namely, rectangles and ellipses. Creating or modeling a shape by boolean set operations is a quite traditional method in both 2D and 3D graphics, and many commercial graphic applications have been adopted this technique. Generally, in these applications, users should select multiple target shapes first, and then execute a set operation. This procedure is based on an idea called noun-verb construction[7].



**Fig. 2.** An example of set operation in a general 2D drawing system

For example, if you want to draw a shape as shown in Fig. 2, the shape can be obtained by the following steps:

1. Select the ellipse and the large rectangle.
2. Execute subtract command.
3. Select the new shape and the small rectangle.
4. Execute add command.

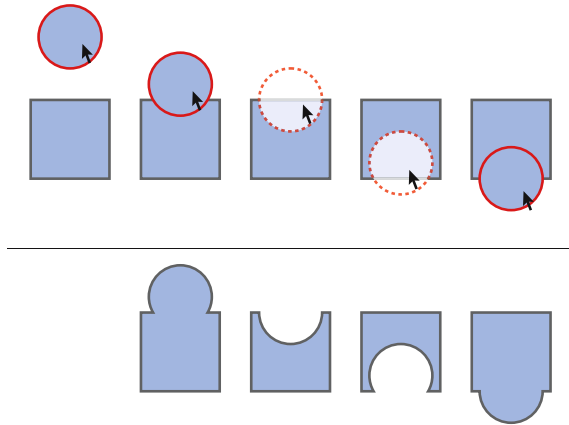
Like this example, in previous systems, users need to select all target shapes at each step, and this procedure makes drawing tasks more complicated and tiresome.

In order to lessen users' drawing steps, our system is designed by the following policy: "The target of an interactive operation is limited to only one shape, and users proceed a task by manipulating the target shape and some related shapes." Therefore, there is no multiple selection mode in this system. We call this approach *Direct Manipulation of Primitives*.

**Geometric Primitives.** Rectangles and ellipses are supported as geometric primitives in this system. A user first selects the rectangle tool or the ellipse tool from the tool palette, and specifies two diagonal points by dragging a mouse. The drawing interface is a very general way to draw this kind of primitives in many drawing system. Drawn shapes can be moved, resized and rotated by mouse operations.

**Set Operation.** The target shape and some other shapes which overlap the target shape are the operands of a set operation. Each shape has an attribute of either adding shape or subtracting shape, and add shapes are painted opaque with solid boundary, while subtract shapes are translucent with dotted boundary as shown in Fig.3 top.

During the drawing process, users need to switch the attribute of shapes. We propose an interface to switch the attribute without any special commands or modes. For example, to switch an adding shape to a subtracting shape, a user selects the adding shape as the target shape, and moves it by dragging. Then, when the mouse pointer enters another shape, the target shape turns into a subtracting shape. On the other hand, when a user moves the subtracting shape which overlaps another shape A, by moving the mouse pointer from the inside of shape A to its outside, the target shape turns into an adding shape again.



**Fig. 3.** The transition of shape's attribute while dragging and results of set operation at each step.



**Fig. 4.** Rounding corners by placing tangent ellipses

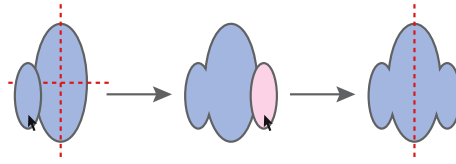
Figure 3 top shows the transition of the target shape by dragging it across another shape, and the bottom row is the results of set operations at each step.

This interface is based on an idea that adding shapes implies pushing a shape out from another shape, and subtracting shapes implies denting a shape from outside. So it may be easy for users to understand the behaviors of this interface.

**Rounding Corners.** Although set operations are simple and useful feature, it is not enough for drawing a variety of shapes. Our system provides a corner-rounding operation which can be used in a similar way to set operations. As shown in Fig. 4, when a user places the target shape of an ellipse to a corner so that the ellipse is tangent to the neighboring edges and double-clicks the target shape just like the execution of a set operation, the corner is rounded. This operation is applicable to both convex and concave corners.

## 2.2 User Support Based on Geometric Constraints

The system supports users by checking geometric relationships between the target shape and other existing shapes. If the target shape is likely to fulfill some geometric conditions or constraints during the drawing task, the system tries to satisfy the conditions, or to suggest users to satisfy them. Our first system supports the following geometric constraints:



**Fig. 5.** The suggestion of a line symmetrical candidate

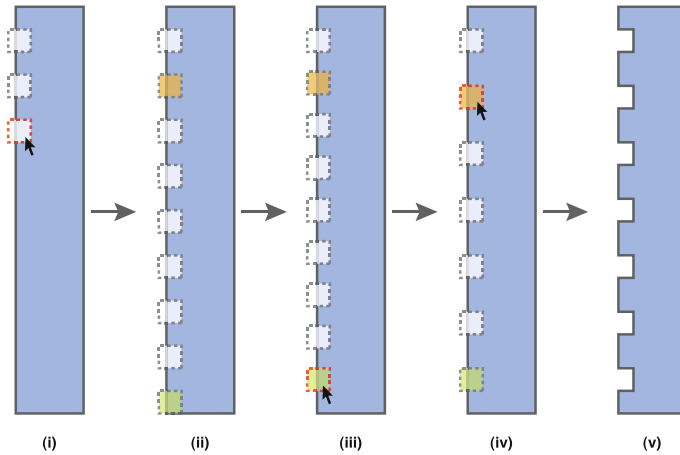
- location and size of the shape,
- line symmetrical placement of the shapes.
- iterative placement of the shapes.

The system always checks these geometric relationships between the target shape and other shapes. Just like set operations or a corner-rounding operation, users need not select multiple shapes in advance to check relationships.

**Location and Size.** This feature is very similar to the concept of Snap-Dragging. If any one of 1) minimum  $x$ -coordinate value, 2) maximum  $x$ -coordinate value or 3) center  $x$ -coordinate value of the target shape is nearly equal to any of 1), 2) or 3) of another shape, the system snaps the target shape. The system checks the  $y$ -coordinate as well. Besides, when any one of 1) height of the target shape, 2) its width, 3) length of a line segment of its components, 4) width of the underlying ellipse, or 5) height of the underlying ellipse whose arc segment consists the shape is nearly equal to any of 1) ~ 5) of another shape or the target shape itself, the system snaps the target shape so that those values are identical.

**Line Symmetrical Placement.** When the target shape is line symmetrical with respect to a line which is parallel to the  $x$ -axis or  $y$ -axis, the shape is drawn with its axis of symmetry on a canvas. If a user performs an operation to a line symmetrical shape, the system displays the same shape as the target shape at the opposite side of the symmetrical axis as a candidate of a new target shape as shown in Fig. 5. To perform the new set operation, the user just needs to single-click the candidate target. This suggestion is made after both set operations and corner-rounding operation.

**Iterative Placement.** If a user wants to place shape A's evenly spaced on a segment of shape B, the user duplicates shape A twice and places them on the segment as shown in Fig. 6 (i). Then the system continues to copy and place the shapes until the segment ends and highlights the second shape and the last shape with different colors (ii). The location of the last shape defines where the iteration ends (iii), and the position of the second shape controls the interval of the iteration (iv). Once the user fixes these parameters and double-clicks any one of the shapes, a set operation is performed (v). Note that this suggestion is



**Fig. 6.** An example of an iterative placement suggestion

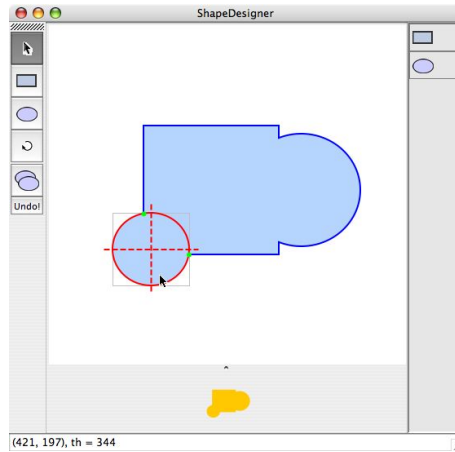
made not only aligning shapes onto a segment, but also shapes on an ellipse. In that case, only adjusting the number of iteration is available since there is no need to adjust the endpoint of the iteration.

### 2.3 User Tests and Results

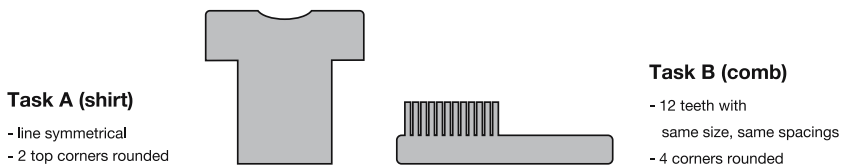
We implemented a prototype system based on the framework as we described above, and did some user tests using our prototype system and a standard drawing system in order to compare the difference of performance. Figure 7 is a screenshot of the prototype system. We used Adobe Illustrator CS (version 11.0) as a standard drawing system, and an optical mouse as an input device. All actions of test users on the screen are recorded by a camcorder, and time to finish the task as measured.

Users had to draw two different types of shapes shown in Fig. 8 using both the prototype and standard systems. First of all, instructions of how to use both systems are made by the experimenter. Since the standard system has an enormous number of operations, the instruction is limited to the operations which seem to be necessary to complete the tasks (for example, drawing and editing rectangles and ellipses, aligning shapes, set operations, etc.). Users are presented the following conditions. 1) Complete the task as quick as possible under the geometric constraints as specified in Fig.8. 2) The nearly same shape is acceptable. 3) The test user may give up the task at any time.

Seventeen students participated in this test. Only four users have an experience of commercial drawing applications such as Illustrator, although all users are familiar with using computers based on graphical user interface. In order to avoid the learning effect, half of the participants used the prototype system first, while the other half participants began with the standard system.



**Fig. 7.** A screenshot of the prototype



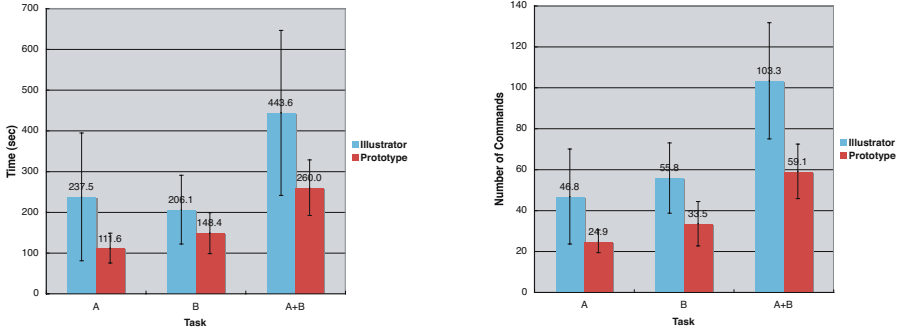
**Fig. 8.** The task shapes used in the user test

**Drawn Results.** In the test, no one gave up the task and every users drew shapes satisfying all the constraints. There is no results which is extremely different from the original shapes. Thus, both systems are useful enough to complete the tasks for non-expert users.

**Completion Time.** The average completion time of tasks is shown in Fig. 9 left. Welch's test showed that all tasks are completed significantly faster by using the prototype system rather than the standard system ( $p < 0.01$  in task A,  $p < 0.05$  in task B, and  $p < 0.01$  in task A+B).

**Number of Commands.** By analyzing the recorded video, we counted the number of commands during the users' drawing process. We define any of the following five actions as a *command*: 1) a single-click of the mouse button, 2) a double-click of the mouse button, 3) mouse drag, 4) a menu selection, and 5) key pressing. Like a menu selection with keyboard shortcut, if a single action corresponds to several types of commands, it is counted as one command. Figure 9 right shows the average number of commands while performing the tasks. Same as the completion time, the prototype system needs significantly less commands





**Fig. 9.** The average completion time and number of commands for Task A, B and A+B. Error bars indicate standard deviations.

to accomplish the tasks ( $p < 0.01$  in task A,  $p < 0.001$  in task B, and  $p < 0.001$  in task A+B).

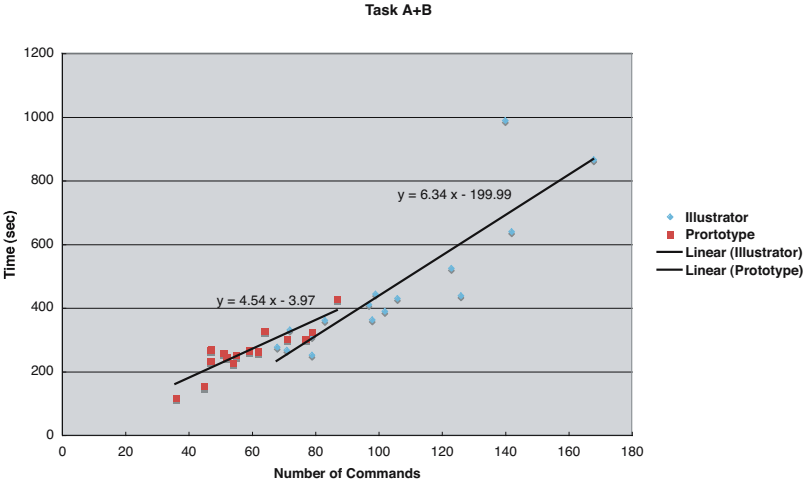
Figure 10 shows the relationship between the completion time and number of commands. The correlation coefficient is 0.89 for both the prototype and standard systems, and 0.92 if samples of both systems are treated as one data set. Therefore, we may say that there is linearity between time and commands on whichever system, so less actions leads to shorter completion time.

In addition to these results, we asked the users which system is easier to use. Fifteen users preferred the prototype system, one user the standard system, and one user answered that there is no significant difference.

### 3 Seamless Mode Transition

While the first prototype system based on direct manipulation of primitives and user support by geometric relationships made certain contribution to reducing time and commands, still some problems on usability were found through the user test. For instance, users accidentally generate tiny shapes by clicking or dragging a mouse in a moment. This kind of garbage shapes can be avoided by checking the sizes of generated shapes.

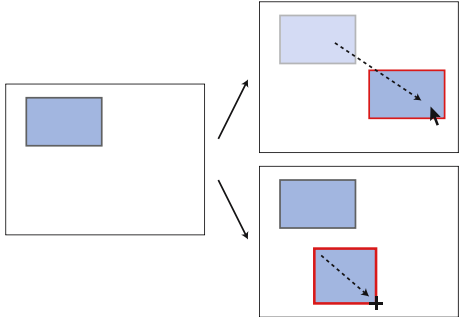
The more serious problem is that the users still forgot to change modes before performing actions though the prototype system has a quite limited number of modes. We eliminate the mode-changing commands of clicking the tool palette to avoid this type of errors. Instead, a new way of switching modes is introduced which is based on the context of the mouse pointer location. That is, as shown in Fig. 11, selection mode is activated when the mouse pointer is located inside any existing shape, while drawing mode is activated when it is located in the background of the canvas. This method is partly similar to the way of integrating multiple actions in Honda's system or an auto-selection mode in Adobe Photoshop which selects a layer containing the first object clicked by mouse. It may



**Fig. 10.** The relationship between completion time and number of commands at Task A+B

reduce both mode-switching errors and number of commands by dismissing the mode-switching command. But as a trade-off, users are not able to draw a shape by dragging a mouse pointer from any points inside an existing shape.

However, there is no way to change the drawing primitives. To handle multiple type of primitive shapes, we propose a new mouse action for switching the type of primitives. Figure 12 illustrates an example. 1) A user draws a shape by simply dragging a mouse. 2) If the user wants to change the primitive (from a rectangle to an ellipse in Fig. 12), the user drags the mouse to the opposite side of the starting point. 3) The user drags the mouse to the first-dragging side again, then the drawing primitive is switched. 4) Repetition of this action toggles the primitive types. This technique can be categorized as a kind of mouse gesture interpretation such as SKETCH[8] system.



**Fig. 11.** Seamless mode transition. Tools are switched automatically according to the mouse location.

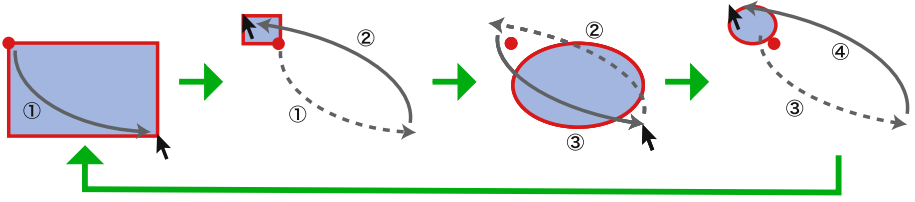


Fig. 12. Switching shapes by mouse dragging

### 3.1 Prototype System

We implemented the second prototype system by modifying the first system to experiment the ideas described above. Figure 13 is a screenshot of the system. Buttons for switching modes are removed, and only four commands (duplicate, delete, undo, and save) are assigned to the buttons on the toolbar. Drawn shapes can be saved as a SVG format and reedited by other graphic applications.

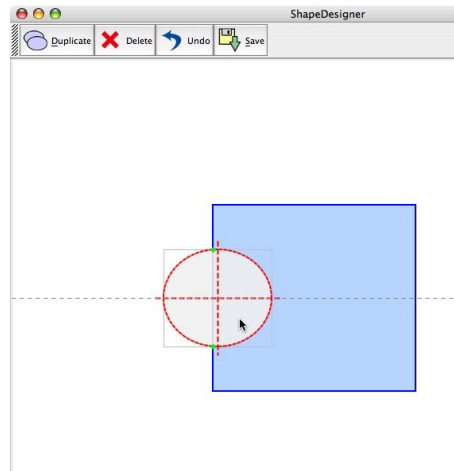
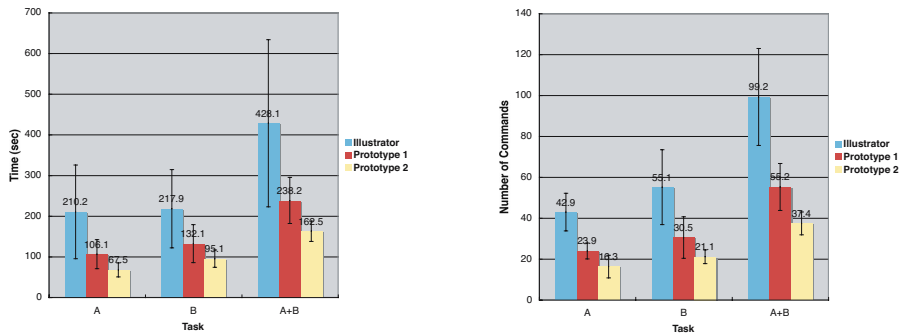


Fig. 13. A screenshot of the second prototype (trimmed)

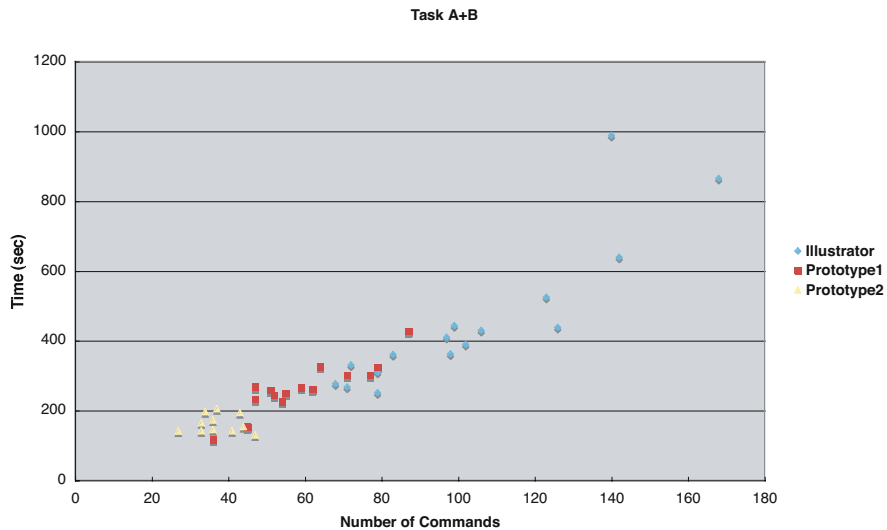
### 3.2 User Tests and Results

Again we did a user test to compare the performance of two prototypes<sup>1</sup>. The test tasks and procedure is the same as the previous test, except for two points. Firstly, eleven of eighteen users who participated in the first test joined the second test. Secondly, only the second prototype system is tested since the first prototype and the standard systems are already tested by the same users. In this section, we call the first prototype system P1, and the second system P2.

<sup>1</sup> A sample interaction movie is available at  
<http://www.graco.c.u-tokyo.ac.jp/~yohki/md/sample.mov>



**Fig. 14.** The average completion time and number of commands of Task A, B and A+B. Error bars indicate standard deviations



**Fig. 15.** The relationship between time and commands

**Results.** The average completion time of the tasks and number of commands are shown in Fig. 14. Results of Welch’s test of completion time are  $p < 0.01$  in task A,  $p < 0.05$  in task B, and  $p < 0.01$  in task A+B. Those of commands are  $p < 0.01$  in task A,  $p < 0.01$  in task B, and  $p < 0.001$  in task A+B. Both results show that the tasks using P2 are accomplished with significantly shorter time and less commands.

The relationship between the completion time and number of commands is plotted in Fig. 15. The correlation coefficient is 0.92, which still shows the linearity between time and commands. We carefully examined the tasks and

found the minimum number of steps to complete the tasks. The ratios of the average number of commands to the minimum number of commands are 1.53 and 1.44 in P1 and P2, respectively. This indicates that there is no significant difference of error ratio. Therefore, the reduction in completion time is mainly due to that of commands by modeless drawing. It seems the error preventing features did not significantly contribute.

## 4 Conclusion and Future Work

In this paper, we described a new simple interface for 2D drawing to lessen user's confusion and steps of operations. Through the user tests, we confirmed that our system contributed to shorten the completion time and reduce the number of commands.

However, there are also some drawbacks on the proposed system. Firstly, the type of shapes is limited. For example, shapes containing curves such as Bezier or B-Spline curves cannot be drawn. Shapes composed of primitives other than rectangles or ellipses such as triangles are difficult to draw because only two types of primitives are available. The problem of what and how primitives or constraints to be handled needs further discussion. Secondly, as a result of disabling the toolbar, users may have a little difficulty in understanding the provided modes or features in their first experience. However, necessary instruction will be quite short and users will be get used to this system quickly.

## References

1. Bier, E., Stone, M.: "Snap-Dragging", *Proceedings of SIGGRAPH '86* (1986) 233–240
2. Gleicher, M.: "Briar: A Constraint-Based Drawing Program", *Proceedings of CHI '92* (1992) 661–662
3. Honda, M., Igarashi, T., Tanaka, H., Sakai, S.: "Integrated Manipulation: Context-aware Manipulation of 2D Diagrams", *Proceedings of UIST '99* (1999) 159–160
4. Igarashi, T., Matsuoka, S., Kawachiya, S., Tanaka, H.: "Interactive Beautification: A Technique for Rapid Geometric Design", *Proceedings of UIST '97* (1997) 105–114
5. Masui, T.: "HyperSnapping", *Proceedings of the Symposia on Human-Centric Computing – Languages and Environments* (2001) 188–194
6. Raisamo, R.: "An Alternative Way of Drawing", *Proceedings of CHI '99* (1999) 175–182
7. Raskin, J. *The Humane Interface*, Addison-Wesley (2000)
8. Zeleznik, R., Herndon, K., Hughes, J.: "SKETCH: An Interface for Sketching 3D Scenes", *Proceedings of SIGGRAPH '96* (1996) 163–170

# Tentative Results in Focus-Based Medical Volume Visualization

Timo Ropinski, Frank Steinicke, and Klaus Hinrichs

Institut für Informatik, Westfälische Wilhelms-Universität Münster, Germany  
ropinski@math.uni-muenster.de, fsteini@math.uni-muenster.de,  
khh@uni-muenster.de

**Abstract.** We describe our current work on volume visualization techniques developed with the aim to enhance interactive exploration of medical datasets. Volumetric lenses can be applied to a volume dataset to interactively focus regions of interest within these datasets. During rendering the parts of a volume dataset intersecting the lens, which is defined by a convex 3D shape, are rendered using a different visual appearance. The lenses proposed allow to apply non-photorealistic rendering techniques interactively to aid comprehension for medical diagnosis.

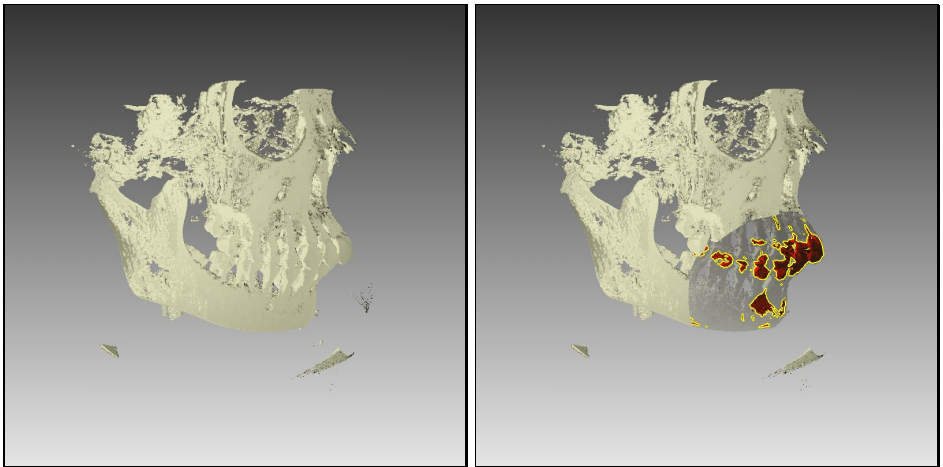
## 1 Introduction

Visualization of medical volume datasets provides an essential diagnosis tool for medical applications. With the availability of powerful graphics hardware it became even more and more important over the last years. Today, medical volume rendering techniques ease health professionals' work during medical diagnosis. Due to the rapid development of high precision medical imaging techniques, e.g., CT, MRI and PET, the spatial resolution and with it the amount and complexity of data increases. Therefore medical diagnoses become more and more challenging. Since in most medical applications the region of interest, e.g., tumors or arterial structures, is small in comparison to the overall dataset, mechanisms are needed which support health professionals to focus on these regions. Furthermore, certain features and properties of anatomical tissues are essential for the diagnosis and need to be preserved, e.g., size and shape of pathologies as well as their spatial position and vicinity to other anatomical structures. Therefore, it is important that visualization techniques consider these demands and aid comprehension by visualizing contextual structures and object relations.

We exploit an algorithm that allows to determine regions of interest within a dataset interactively, to which a different visual appearance is applied [3]. The region of interest, sometimes referred to as lens, is given by an arbitrary convex shape and a visual appearance associated with it. To be applicable to volume data we have extended the algorithm. Thus, volume data intersecting the lens can be emphasized using arbitrary rendering techniques, e.g., edge-enhancement, isosurface rendering etc.

## 2 Interactive Focussing Using Focus and Context Visualization

Medical visualization can be improved by using NPR techniques [2]. The lens described in this section has been developed with the goal to provide the user insights into datasets by still preserving spatial cues, and to support the user to determine the spatial relationship to the rest of the dataset during exploration. In Figure 1 (right) a lens is applied to a CT scan of a human skull, i.e., the skull is rendered using isosurface shading outside the region of interest while special rendering techniques are applied within the region of interest, whereas shaded isosurface rendering with no lens applied is shown in Figure 1 (left). Shaded isosurface rendering is a good mechanism to provide the user with cues about the overall structure of objects and their spatial relationships. It does not contain as much information as for example direct volume rendering (DVR) where several voxels contribute to one pixel, but it provides a better overview of the surface structure. However, due to the fact that only one voxel contributes to each pixel's color, isosurface rendering does not allow to achieve insights into a dataset. Therefore, the lens is applied to allow the user to analyze the inside structure of the dataset by still maintaining the contextual information of the isosurface rendering outside the region of interest. In the sample rendering of the skull inside the lens the inner structure of the teeth is rendered using isosurface shading combined with a different transfer function. To draw the user's attention to these structures their silhouette is highlighted with a boundary. Furthermore rendering the front faces of the teeth translucently we ensure that the user can perceive the spatial relationship of the inner structure. Although several tech-



**Fig. 1.** CT scan of a human skull. Isosurface shading (left); spherical lens applied giving an occlusion free view to the highlighted inner structures. To provide a better focussing on the region of interest the voxels lying behind the lens have not been rendered (right).

niques are combined within the lens, there is no significant loss of performance using the describe approach compared to other more simple lens styles. This is ensured by processing the section of the ray intersecting the lens only once during the main rendering pass. Thus the three different visual appearances, i.e., translucent rendering, highlighting and isosurface shading, can be combined in a single rendering pass in a front to back order by using the following blending equations for determining the color resp. alpha value:

$$C_{dst} = C_{dst} + (1 - \alpha_{dst})\alpha_{src}C_{src} \quad (1)$$

$$\alpha_{dst} = \alpha_{dst} + (1 - \alpha_{dst})\alpha_{src} \quad (2)$$

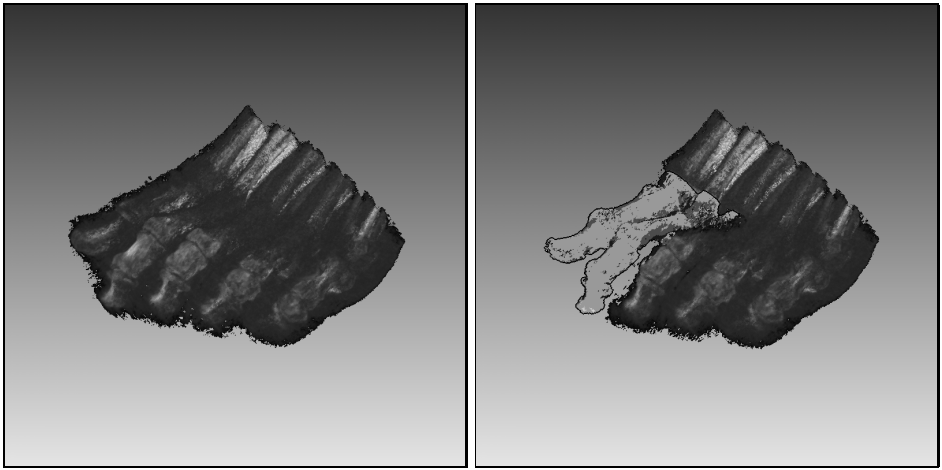
$C_{dst}$  denotes the RGB components of the destination color, while  $C_{src}$  denotes the RGB components of the source color and  $\alpha_{dst}$  resp.  $\alpha_{src}$  denotes the alpha value of the destination resp. source pixel. Only for the edge detection based highlighting a preprocessing pass is needed. In this pass parts of the volume lying inside the region of interest are rendered as non-shaded isosurfaces; in Figure 1 (right) an appropriate isovalue which can be used to extract the inner structure as object of interest has been determined interactively. The result of this preprocessing pass is stored in a framebuffer-sized texture which is accessed during the main rendering pass, in which the edge detection required for highlighting objects of interest is performed as initial step. Therefore the previously created texture is accessed four times at the neighboring texels corresponding to the current pixel position. The fetched color values  $c_{east}$ ,  $c_{north}$ ,  $c_{west}$  and  $c_{south}$  are used to determine whether the current fragment belongs to an edge in image space:

$$fragcolor = |(c_{east} - c_{west}) + (c_{north} - c_{south})| \quad (3)$$

If *fragcolor* contains the maximum intensity in all three channels, i.e., it is white, the current fragment belongs to an edge in image space. In this case the current fragment color is set to the desired color used for highlighting objects of interest. By considering the different color channels R, G, B and A separately the described edge detection mechanism can be easily extended to support highlighting of different objects without needing extra rendering passes.

After the edge detection has been performed, in those cases where the current fragment does not belong to an edge the volume dataset is processed. Therefore, a ray is cast through the parts of the volume which intersect the lens. To achieve the visual appearance shown in Figure 1 (right) an isosurface rendering mode is used to determine the front faces rendered translucently. In cases where the ray intersects the volume, instead of terminating the ray after the isovalue has been encountered on the ray the ray is further processed until an object of interest, i.e., the inner structure, or the farthest back face of the volume dataset is hit. In both cases an appropriate shading is applied, which ensures that the back face is rendered translucently and the object of interest with the corresponding transfer function. Since the ray is further processed after a voxel contributing to the visualization is encountered, the number of samples processed per ray is not larger than when applying DVR inside the lens.





**Fig. 2.** A rotational b-plane x-ray scan of a human foot. DVR (left); cuboid toon-shading lens applied to reveal the bone structure.

Another medical example application of the lens metaphor is illustrated in Figure 2, which shows a lens applied to an x-ray scan of a foot. The foot is rendered using DVR outside the region of interest to provide an overview of the interior structure. Inside the voxels' gradients are exploited and toon-shading [1] is applied to the extracted isosurfaces of the bones. The toon-shading approach enhances depth perception, which results in improved comprehension. Furthermore to let the bones become more apparent a black outline is rendered using the edge detection technique described above.

The two presented visualization techniques are application of our current work on focus-based medical volume visualization. Based on this work we are developing visualization techniques specialized to support interactive exploration of PET datasets.

## References

1. Amy Gooch, Bruce Gooch, Peter Shirley, and Elaine Cohen. A Non-Photorealistic Lighting Model for Automatic Technical Illustration. *Computer Graphics*, 32(Annual Conference Series):447–452, 1998.
2. Aidong Lu, Christopher J. Morris, David Ebert, Penny Rheingans, and Charles Hansen. Non-Photorealistic Volume Rendering using Stippling Techniques. In *VIS '02: Proceedings of the conference on Visualization '02*. IEEE Computer Society, 2002.
3. Timo Ropinski and Klaus Hinrichs. Real-Time Rendering of 3D Magic Lenses having arbitrary convex Shapes. In *Journal of the International Winter School of Computer Graphics (WSCG04)*, pages 379–386. UNION Agency - Science Press, 2004.

# 3d Visualisation in Spatial Data Infrastructures

Torsten Heinen<sup>1</sup>, Martin May<sup>1</sup>, and Benno Schmidt<sup>2</sup>

<sup>1</sup> Institute for Geoinformatics, University of Münster, Germany

{heinent, mmay}@uni-muenster.de

<http://ifgi.uni-muenster.de>

<sup>2</sup> con terra GmbH, Münster, Germany

[schmidt@conterra.de](mailto:schmidt@conterra.de)

<http://www.conterra.de>

**Abstract.** 3d visualisation of real spaces in the Internet is nowadays getting ready for all day usage. Because of higher complexity of 3d- compared to 2d visualisation, it is mostly used for high-end visualisations. The strategy of integrating them into existing spatial data infrastructures and making reuse of existing resources that are so far mostly used for 2d visualisation, created the opportunity to build 3d web-applications as an add-on. The recurring problems of web-based 3d visualisation raised the question, how special modules for 3d visualisation could also be made reusable. We took a closer look especially on the so-called mapping-level, which is essential for the type of visualisation, interactivity and some special analysis-tasks.

## 1 Introduction

At the moment there is a rapid development in Web-mapping and geoinformation via the Internet. Structures providing access to such geoinformation are called spatial data infrastructures (SDI). These SDIs replace former approaches in working with geoinformation from local management and processing of data sets to distributed systems. One important aspect for future usage and reliability is interoperability and standardisation. Cross-system and cross-institutional interoperability is achieved by utilisation and continuous further development of standards, which are internationally accepted and mainly organised by the Open Geospatial Consortium (OGC) and the International Standardisation Organisation (ISO). To meet the requirements of all-day-usage the specified services must not restrict to provide data, but also have to make several functionalities accessible.

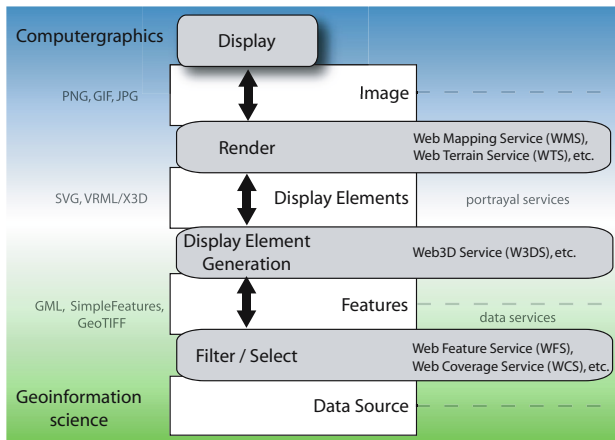
For better understanding of 3d visualisation in geo-space (3d-geovisualisation) we have to differentiate between the involved software-domains: the geoinformation science on the one hand and computer graphics on the other (Schmidt 2002). This transformation process is one of the most fundamental tasks in 3d geovisualisation and bridges the gap between these two domains, taking into account the strength and weaknesses of both.

This paper presents a more fine-grained approach towards a service-oriented 3d geovisualisation architecture by offering a loosely coupled web service for

the transformation from geospatial data to display elements. This service, called *VsgMappingService* (VMS), is based on the workflow-managed chaining pattern (Percivall 2002) and produces a virtual scenegraph (VSG) of geospatial data. Common SDIs provide access to geodata with the help of standardised geoinformation (GI) services. Based on the virtual scenegraph, this mapping service can output different scenegraph formats, like VRML<sup>1</sup>, X3D<sup>2</sup> or Java3d<sup>3</sup>, which then can be used by service consumers (that means other services or client applications) to render static images or realtime 3d graphics.

## 2 Service-Based 3d Geovisualisation

While in today's SDIs the mapping of geoinformation focuses mostly on 2d visualisation, several GI-services can also be used for creating 3d geovisualisations. Several younger initiatives evolved some of the mandatory elements for the software-domain of 3d geovisualisation, which now gets web- and standard-based and builds upon existing SDIs (Schmidt et al. 2003). Services can be categorised according to the portrayal model (Buehler 2003) as shown in Figure 1.



**Fig. 1.** Portrayal pipeline, associated standard formats (left) and services (right)

The portrayal model is based on the concept of the visualisation pipeline (Haber & McNabb 1990) and describes the processes (grey boxes) to filter and transform non-graphical data to graphical representations. These representations can then be rendered and displayed on different output devices. In contrast to SDIs, traditional geoinformation-systems (GIS) provide all processes in a single monolithic application. In earlier work we showed combinations of different weighted service-based client-server architectures (Schmidt et al. 2003).

<sup>1</sup> see <http://www.w3.org/hypertext/WWW/MarkUp/VRML/> for more information.

<sup>2</sup> see <http://www.web3d.org> for more information.

<sup>3</sup> see <http://java.sun.com/products/java-media/3D> for more information.

Sharing of geoinformation in spatial data infrastructures takes place through standardised web services. For example does the *Web Feature Service* (WFS, Vretranos 2002) and *Web Coverage Service* (WCS, Evans 2003) enable the access to discrete geographical data (features) and values or properties of a set of geographic locations (coverages) respectively. Unlike these data services, portrayal services do not allow the retrieval of spatial data but only pictures or maps of the data. In case of *Web Mapping Services* (WMS, Beaujardiere 2004) the images contain 2d views on the data. The *Web Terrain Service* (WTS, Liebermann & Sonnet 2003) can be seen as a first step towards service-based 3d visualisation as it generates rendered 3d views of spatial data and sends it to the service consumer in common image formats. The *Web3d Service* (W3DS, Quadt & Kolbe 2005) tries to bridge the gap between the access of images of data and raw geospatial data: The W3DS is a portrayal service for three-dimensional geodata and delivers display elements.

Note that most implementations of portrayal services include local data sources and therefore are also responsible for filtering ('filter/select') and transforming spatial data into graphical representations ('display element generation'). The type of representation can be additionally determined by *Styled Layer Descriptors* (SLD) in 2d as well as in 3d. This might be realised by future versions of the VMS.

### 3 Motivation

3d portrayal services are a first step towards service-based 3d visualisation. But current specifications and implementations are, from the user or client perspective, tightly coupled to other services or data. In contrast to tightly coupled services, a loosely coupled service provides their functionality independently from further services or data sources (Percivall 2002). While data services need to be tightly coupled, other service should offer their functionality on a loosely coupled basis (Erl 2004).

Neither the current WTS nor the W3DS specification allows dynamic binding of third-party data sources yet. Although the service provider might add or remove data sources and even use other services (aggregate or opaque chaining, Alameh 2002), the user has to choose static data layers. In addition to the lack of selecting third-party data, the user does not have any influence on the way geodata are mapped to display elements.

To overcome these problems, this paper presents a prototypical architecture for an interoperable and loosely coupled mapping service that isolates the step of mapping geoobjects to visualisation-objects. This way the mapping service is more fine-grained than the W3DS and allows control over the mapping processes as well as the used data source.

**Scenario.** Planner A is assigned to extend a road network. He wants to interactively explore the region with a new endangered species dataset from agency X and his own street network data from agency Y. In addition, he likes to integrate his new data with a specific 3d model service from agency Z and height

information from agency W. After interactively exploring the region in 3d, he is assured that the extension of the road network will endanger a region worth of protection. To convince the project partners to choose another way for the road, he likes to produce a fly-over movie. In his final report he also likes to include a 3d map of the surrounding area.

This scenario shows that different data sources need to be mapped to different representations, which in turn need to be rendered to different output media on several client-types (browser or standalone application).

## 4 Concept

The review of existing solutions for visualisation in the web shows that the mapping is mostly realised as an integrated step in a monolithic service. Focussing on 3d visualisations it seems to be self-evident that this might be an opportunity to reuse large pieces of software.

Our concept follows a more fine-grained approach as outlined by the visualisation pipeline. It clearly separates functionalities of (geospatial) data access, mapping, and image generation (rendering). While geospatial data access is mostly standardised and implemented in SDIs, we focus on the transformation of spatial data into 3d representations.

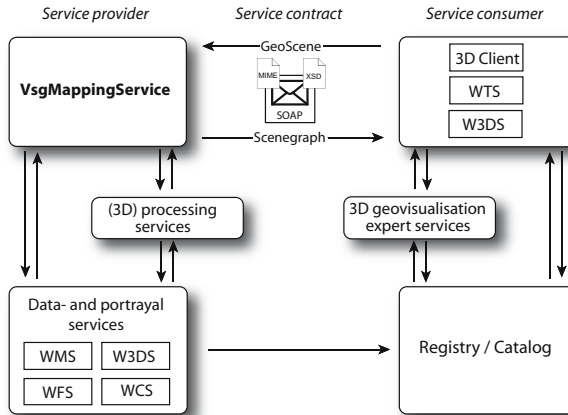
Following the design-approach to separate functionalities and to abstract the mapping-level by the creation of interfaces (Fowler 2002) we have to keep in mind the overall design-guideline: 'keep it simple'. It seems as if the modules get smaller, the complexity of the overall application increases. To quantify the achievement of objectives we will have to make use of software-metrics which includes statement, expression, and data complexity. The appropriate methodology for this future work seems to be described in (Rechenberg 1986).

### 4.1 Architecture

The integration of the mapping service into common SDIs is illustrated in Figure 2. Building upon existing data- and portrayal services as well as special processing services, the *VsgMappingService* communicates with service consumer using the *Simple Object Access Protocol*<sup>4</sup> (SOAP), which has lately been under examination by OGC (Sonnet & Savage 2003). The service contract is defined using XML Schema<sup>5</sup>. The orchestration of different GI-services is managed by the *VsgMappingService* using the translucent or workflow-managed chaining pattern. The service consumer (e.g., an interactive 3d client or another service) has influence on the execution of the workflow by specifying different parameters in the service request. These parameters include the type of the *GeoScene* description (see next chapter), the used data sources and scene-specific settings. The service consumer is responsible for the selection of appropriate data sources (with proper

<sup>4</sup> see <http://www.w3.org/2000/xp/Group/> for more information.

<sup>5</sup> see <http://www.w3.org/XML/Schema> for more information.



**Fig. 2.** Overview of the proposed 3d geovisualisation architecture

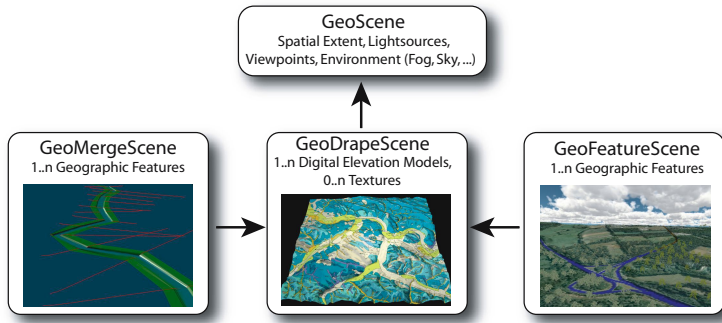
service parameters). The service response is a 3d representation of the spatial data in the form of a scenegraph, which can be used by the service consumer to render the scene on an output device.

## 4.2 GeoScene

*GeoScene* is an abstract, renderer-independent description of 3d geospatial content and is used as the service request for the *VsgMappingService*. This description can be seen as a template which is filled out by the client with information about the location of spatial content (services or URLs) and further settings like environment parameters (fog, light sources, etc.) or viewpoint information. Different types of *GeoScene* (Figure 3) give meaning (semantics) to the transformation process as they define how the mapping from spatial data to the graphical representation should take place.

A river, for example, might be portrayed with a texture placed (draped) on top of the terrain with the use of a simple *GeoDrapeScene*. The source of the texture might be a WMS. An alternative way of visualising and mapping spatial data like a river is provided through the *GeoMergeScene*, which indicates the mapping service to intersect and merge geometry with the digital elevation model. This intersection process might be delegated to a special 'footprint' service (May & Heinen 2004) or calculate the resulting elevation model on the fly with appropriate algorithms. With the use of the *GeoFeatureScene* the mapping service provides ways of adding objects like vegetation, buildings or labels to the 3d scene.

The *GeoScene* description should be clearly differentiated from the scenegraph in computer graphics (Döllner 2000). Whereas the scenegraph is a data-structure, which describes visualisation-elements and their behaviour in rendering-systems, the *GeoScene* specifies some geo-physical parameters which have to be visualised. All coordinates are provided in a spatial reference system

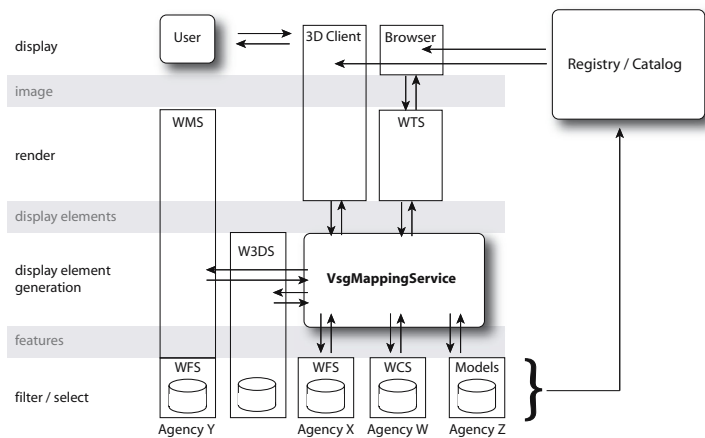


**Fig. 3.** Different types of GeoScene’s and associated components

instead of metric coordinates commonly used in computer graphics. Moreover, the *GeoScene* does not hold the data but rather describes where and how the data can be fetched. This differentiation enables us to visualise the same *GeoScene* (in human terms) in different visualisation-surroundings with totally different visualisation-capabilities.

4.3 Example

Based on the scenario described in section 3 this chapter should briefly illustrate the integration of the VMS into existing spatial data infrastructures (Figure 4). SDIs enable standardised access to geoinformation through data, portrayal and special 3d services, which can be discovered through the use of catalog or registry services. The illustration categorises the components according to the visualisation pipeline and shows their interaction.



**Fig. 4.** Interaction between user, applications and services according to scenario described in Chapter 2

First of all, the user needs to be provided with information about available services and data sources from a catalog service. Then, the user creates an appropriate *GeoScene*. In case of our scenario the following simplified configuration might be suitable: *GeoFeatureScene* with digital elevation model from agency W (WCS), streetnetwork texture from agency Y (WMS), geographic features from agency X (WFS) styled with 3D models from agency Z.

This description contains not only mapping parameters (e.g., output format, level-of-detail, type of geometry, etc.) and service locations but also service specific parameters, like cell size for the WCS elevation data. As the *GeoScene* is platform-, media and render independent, it can be used in either standalone client applications or different services (e. g. WTS or W3DS) to invoke the *Vs-MappingService*. The VMS is then responsible for executing the workflow and transforming the spatial data into 3d representation. Finally, these representations can be rendered by the client or 3d portrayal service to present the final output to the user.

## 5 Conclusion

This work presents a component that encapsulates the fundamental mapping-level of web-based 3d geovisualisation. Example-applications (interactive 3d clients and WTS) have been realised to show the reusability of the VMS. Nevertheless, does the reusability has to be proofed also in other contexts and with ongoing development of existing discussion-specifications (WTS, W3DS). Our implementation (and this is somehow a subjective view) reduced the overall codeline-count.

The concept of the *GeoScene* might open a way to realise better adapted 3d-visualisations to user-requirements, because of its technology independency. To close the gap between the semantical description of a *GeoScene* and the technology visualising it, we propose to make use of knowledge-based service-composition. Further research will be undertaken in developing the concept of the *GeoScene* as well as specifying rules which help to create valid and usable service chains for 3d geovisualisations.

Especially the concept of a *GeoScene* will be in focus of further research. It might open a way to realise better adapted 3d-visualisations to user-requirements, because of its technology independency. To close the gap between the semantical description of a *GeoScene* and the technology visualising it, we propose to make use of knowledge-based service-composition. Further research will be undertaken in formalising and specifying rules which help to create valid and usable service chains for 3d geovisualisations.

The proposed concept of an interoperable and loosely coupled web service that provides workflow-managed methods for this transformation task promises various other advantages: reuse of existing gi-services, security of geodata and



rapid development of light-weight 3d-clients and applications. Especially the latter might lead to reduce software development costs for web-based 3d applications.

## References

- Alameh, N.: Service chaining of interoperable geographic information web services. (2002)
- Beaujardiere, J. de La: Web Map Service (WMS). OpenGIS Implementation Specification. (2004)
- Buehler, K.: OpenGIS Reference Model. Approved Technical Baseline 03-040, Open Geospatial Consortium. (2003)
- Döllner, J., K. Hinrichs: A generalized scene graph api. *Vision, Modeling, Visualization* (2000) 247–254
- Erl, T.: *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*. Prentice Hall PTR (2004)
- Evans, J.: Web Coverage Service (WCS). OpenGIS Implementation Specification. Open Geospatial Consortium. (2003)
- Fowler, M.: *Patterns of Enterprise Application Architecture*. Addison-Wesley (2002)
- Haber, R.B., D.A McNabb: *Visualization Idioms: A Conceptual Model for Scientific Visualization Systems*. In: IEEE Computer Society Press (1990) 74–93
- Kolbe Th. H., G. Gröger, L. Plmer: CityGML - Interoperable Access to 3D City Models. In: Oosterom, Zlatanova, Fendel (Eds.): *Proceedings of the Int. Symposium on Geo-information for Disaster Management on 21.-23. March 2005 in Delft*, Springer (to be published)
- Liebermann, J., J. Sonnet: OOG Web Terrain Server (WTS). OpenGIS Implementation Specification 03-081r2, Open Geospatial Consortium. (2003)
- May, M., T. Heinen: Dienstebasierte Aufbereitung von Geländemodellen für die 3d Geovisualisierung. In: *IT-Regionen: Innovation und Technologie als Schlüssel für eine nachhaltige Stadt und Regionalentwicklung*, Wien, M. Schrenk. (2004)
- Percivall, G.: The Opengis Service Architecture (Topic 12). Abstract Specification (ISO19119) 02-112, Open Geospatial Consortium. (2002)
- Quadt, U., T.H. Kolbe: Web 3d Service. Discussion Paper 05-019, Open Geospatial Consortium. (2005)
- Rechenberg, P.: Ein neues Maß für die softwaretechnische Komplexität von Programmen. *Informatik Forschung und Entwicklung*. (1986)
- Schmidt, B.: Verknüpfung der Datenmodelle für GIS und interaktive 3D-Visualisierung. IfGIprints 17. Institute for Geoinformatics / Solingen: Natur & Wissenschaft, Münster (2002)
- Schmidt, B., M. May, C. Uhlenkükken, : Dienstebasierte Architekturen für die Webbasierte 3D-Geovisualisierung. IfGIprints 18. Münsteraner gi-Tage. Institute for Geoinformatics / Solingen: Verlag Natur & Wissenschaft (2003) 311–323
- Sonnet, J., C. Savage: OWS 1.2 SOAP Experiment Report. OpenGIS Discussion Paper 03-014, Open Geospatial Consortium. (2003)
- Vretanos, P.: Web Feature Service (WFS). OpenGIS Implementation Specification 02-058, Open Geospatial Consortium. (2002)

# From Artefact Representation to Information Visualisation: Genesis of Informative Modelling

Iwona Dudek and Jean-Yves Blaise

UMR CNRS/MCC MAP 694 gamsau, EAML 184 av de Luminy,  
13288 Marseille cedex 09, France  
{Iwona.Dudek, Jean-Yves.Blaise}@map.archi.fr  
<http://www.map.archi.fr>

**Abstract.** In the field of the architectural heritage, the representation of artefacts, particularly for communication purposes, has benefited from the development of computer-based modelling techniques in fields ranging from archaeology to geography. But numerous experts in the above mentioned heritage field have come to question the readability of realistic models inside which the hypothetical nature of the content, a reconstruction, is not clearly assessed. In parallel, research in information visualisation has demonstrated that graphics can support reasoning as well as communication. Our contribution introduces the genesis of an *informative modelling methodology* in which the representation of architectural objects is used for information search and visualisation. 3D or 2D models localise objects in time, in space, and in a hierarchy of canonical shapes; they are calculated on the fly and deliver information visually. This paper discusses the underlying modelling methodology and applications in investigations about the evolutions of the city of Kraków (Poland).

## 1 Introduction and Research Background

In the field of the architectural heritage, computer graphics have become an increasingly popular tool for communicating results of historical investigations. Virtual reconstructions are often built in order to let a wide public have an idea of how an architectural object may have been like at time  $T$  of its evolution. But the use of graphics with this sole goal is often discussed (see [1]) in particular on two grounds:

- A lack of readability due to the fact that the inferences for the reconstruction are hidden in the final result.
- An appalling level of usefulness for the researchers themselves who invest time on a reconstruction that in the end remains a side-effect of the research process, giving no access to deeper information levels.

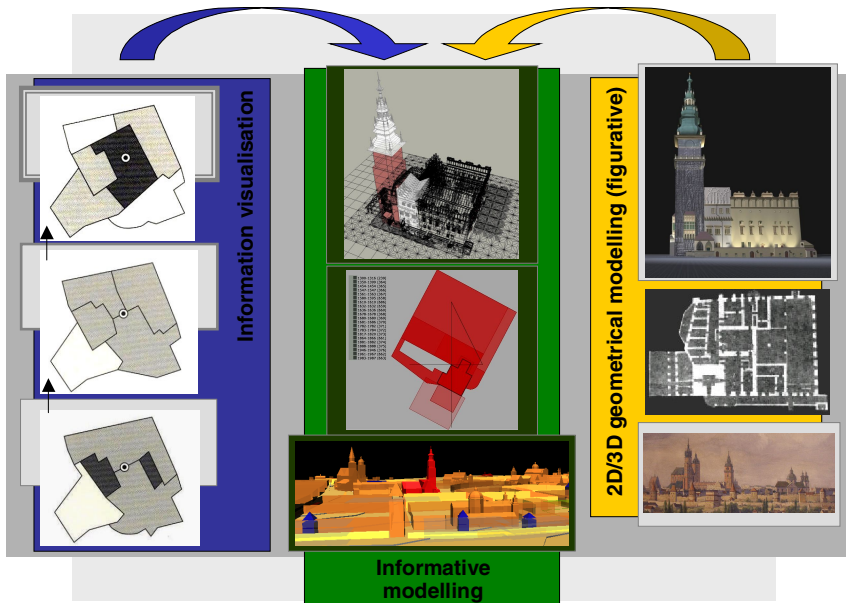
In other words, the information-gathering effort made in order to produce a reconstruction totally evaporates in the final result. The representation is not linked to the sources of information that helped building it, it is not dynamically updated when new findings are done, it does not even mention what at the end of the day seems the most meaningful for researchers - the uncertainty of the original information set.

In parallel, researchers in the field of information visualisation have investigated the use of computer graphics in not only retrieving information but also helping to better sum it up, better understand it. The use of computer graphics as of information discovery tools in the words of J. Kantner [2] is today a reality in many disciplines, but clearly not in the field of the architectural heritage although P. Alkhoven [3] has raised this issue.

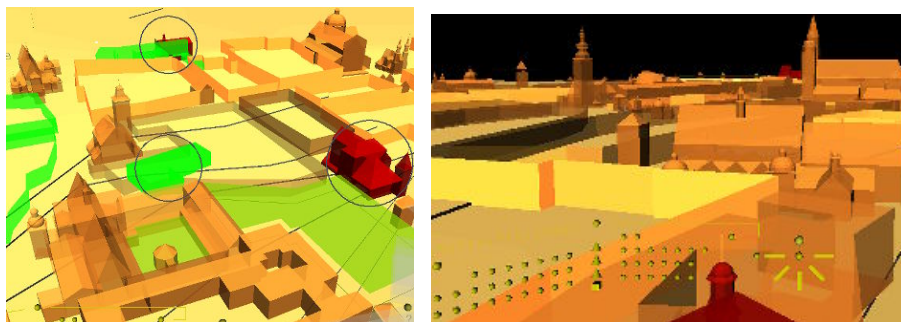
Our contribution is an attempt to try and bridge the gap that one can today observe between the above mentioned fields. We try to lay the basis of basis an *informative modelling* methodology in which the representation of artefacts does not claim veracity, but supports dynamic information retrieval and visualisation.

In the field of the architectural heritage, the objects that we strive to represent are most of the time not fully known. Before an actual reconstruction can be represented, its author works out a reconstruction hypothesis basing on pieces of uncertain information (manuscripts, old maps, etc.) that in almost all cases remain partial (see [4]).

What solution does one have when on one hand he does not really know how an edifice was and when on the other hand its computer tool calls for geometric exhaustiveness and trendy realism? What kind of communication, not to mention reasoning, can be done when such a gap exists between a partial knowledge and a thorough



**Fig. 1.** Filiation of informative modelling. Left, example of insight gained by alternative data distribution accross a territory, from E.R Tufte [5] (graphical analysis of deceased during the London 1859 Cholera epidemic, the circle represents the well finally identified as the source of the disease; only the top distribution indicates this). Right, illustrating the tradition of architectural figurative representation where shape appearances are prominent.



**Fig. 2.** Left, Graphical appearances adapted to VRML: red or blue colouring marks the use of time analogies, emissive colouring stresses information lacks, translucency mark authenticity of the shapes, green highlighting lets users to check what type of documents are available for each object. Scene for period 1775 showing Franciscans church and monastery (foreground), Dominicans, All Saints, saint Joseph, Saint Peter and Paul, Saint Joseph and Michael, Saint Andrew churches, state of knowledge February 2005.

Right; Interactive VRML selectors in the control panel, allowing for instance queries on various BD from each shape; scene for period 1775, state of knowledge February 2005.

geometric description? Naturally this question is a central one inside the community dedicated to historical investigations. In that domain gathering, analysing and understanding the architectural documentation is the core task, and visualising and retrieving that information the real goal of representations (see Fig 2). As a result of the ARKIW research programme (see [4]), we came to consider that the solution may well origin not in the field of 3D modelling but in the field of information visualisation, where the role of graphics as defined by E.R. Tufte [5] obviously better suits our goals: “*We envision information in order to reason about, communicate, document and preserve that knowledge*” [...].

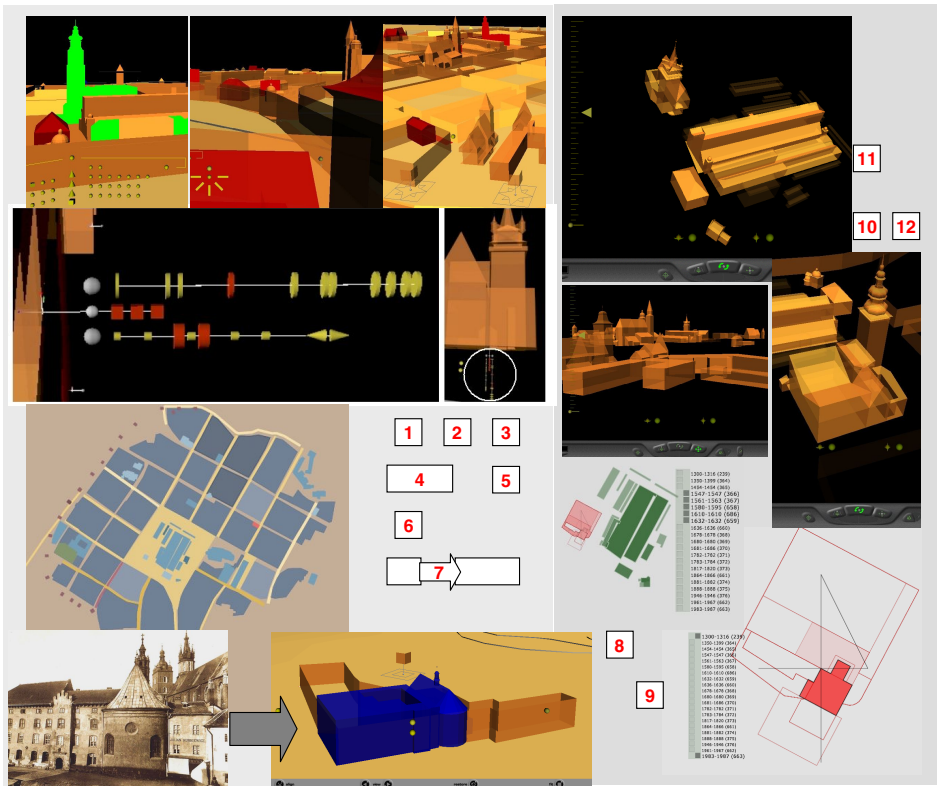
## 2 Principles

During its period of life, an architectural or urban object is in general deeply transformed as a consequence of human activity (ex. adaptations, additions, ..) or of a natural phenomenon (ex. fire, flood, earthquake). We need to document and represent each phase of the edifice’s evolution, and will therefore need to formalise a theoretical model of architectural elements in which each element can be given identity persistence, but state evolutions. This is done by providing a static ontology at concept level and a dynamic ontology at instance level, in the words of [7]. The implementation is based on OOP and introduced in [4].

In the proposition we present, architectural and urban concepts are used as filters on the documentation. Each concept features a group of information (graphical and not graphical) that includes a precise definition of its morphology as well as bibliographical references. Yet the documentation is rarely precise enough to thoroughly document all aspects of a physical object. 3D shapes to which we will want to attach pieces of information may then be incompletely defined, and need to be visually

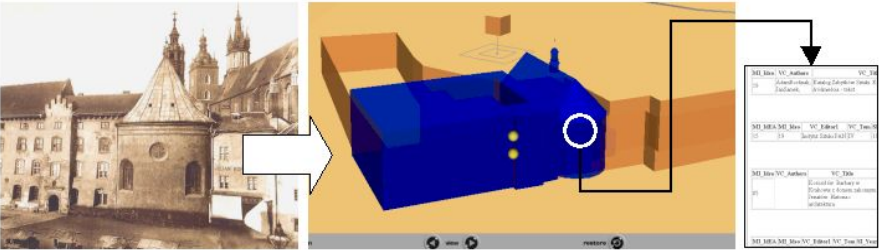
marked with an indication on what information the proposed shape is based on. Briefly said, we face the necessity to provide a dynamic link between three elements:

- Pieces of information, often vague and uncertain. *They will be analysed in order to state at time  $T$  of the research what one can say about an architectural object at various  $T_1$ - $T_2$  periods of its evolution.*
- A hierarchy of architectural concepts at various scales (urban edifice, covering, arch, ...) instanced as a result of the information analysis process. *Each object is natively and dynamically connected to the information sets since representations are calculated as a result of queries on the information sets (see Fig. 4). The representation's level of abstraction and of resolution will match the level of knowledge we have on the object.*
- A set of graphical codes used in order to visualise an evaluation of the nature or accuracy of the documentation attached to each architectural object represented in a scene. Codes are not given once for all but reflect the exact content of the documentation at time  $T$  of the study. *Such codes can show for example the differences between the original parts and elements that were added later; or help in distinguishing what is certain in a hypothesis and what is only hypothetical.*



**Fig. 3.** Alternative types of graphics: 1,2,3 - type A; 4,5 – type B; 6 – type F; 7 – type E; 8,9 - type G; 10,12 - type C; 11 - type D (see following table).

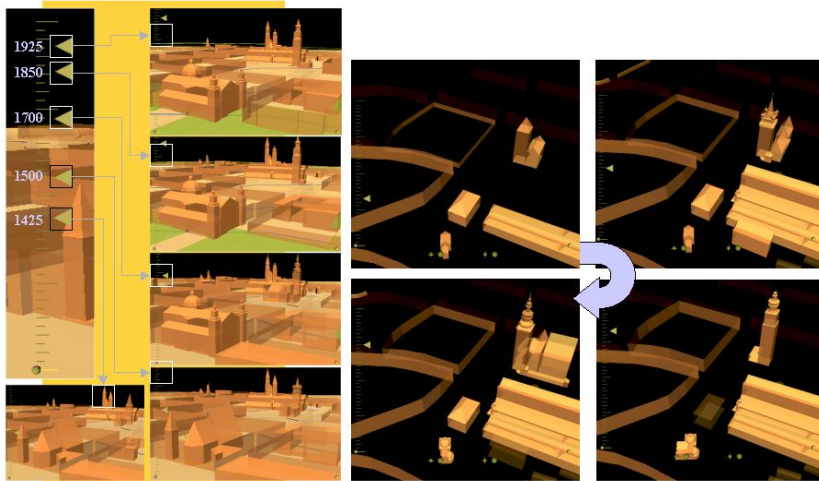
Representation type	Search question to which the representation corresponds
[type A] : Documentation analysis 3D scenes	What <b>documentation</b> do we have <b>on objects of type A,B (...)</b> for <b>period P'</b> [ $T_1$ - $T_2$ ]? Example : “documentation for objects of type Urban Edifice, Urban Block and Fortification Units for period 1790-1791”
[type B] : Typology analysis 3D scenes	What <b>information</b> do we store in the documentation concerning the <b>shape, function and structure on objects of type A,B (...)</b> for <b>period P'</b> [ $T_1$ - $T_2$ ] ? Example : “style, functions and structure of object Sukiennice (Cloth Hall) at date 1550”.
[type C] : Timeline 3D scenes	What are the <b>morphological evolutions of objects of type A,B (...)</b> ? Example : “evolution of objects of type Urban Edifice, Urban Block and Green Area”
[type D] : Object centred Timeline 3D scenes	<b>morphological evolutions of object <math>\alpha</math> showing also surroundings of object <math>\alpha</math></b> , with an additional search criteria that can be instance of the same concept, distance to object, combination of the two previous. Example : “all objects around object town hall in a radius of 200 meters”.
[type E] : Document-content equivalent 3D scenes	Find and represent <b>all the objects quoted in bibliographic/iconographic item <math>\beta</math></b> . Example : step 1 - “all the objects corresponding to I.Krieger’s 1860s photograph”. step 2 - “what other bibliographical items relate to the elements shown on this photograph”.
[type F] : Documentation analysis 2D scenes	What <b>documentation</b> we have <b>on objects of type A,B (...)</b> for <b>period P''</b> [ $T_2$ - $T_3$ ] Example : “ documentation for objects of type Urban Edifice, Urban Block, Fortification Units, Streets and Squares, Green Areas for period 1820-1821”.
[type G] : Object centred 2D scenes with historical layers	<b>morphological evolutions of object <math>\alpha</math>, with the surrounding objects</b> (see D) with interactive 2D historical layers. Example : “all objects around object town hall of type Urban Edifice in a radius of 100 meters”.



**Fig. 4.** Type E representation process, a method to for iterative documentation querying. Left an I.Krieger’s photograph stored as a iconographic item, middle the corresponding 3D scene calculated by selecting the photograph in the DB Web interface, right the bibliographical items corresponding to kościół św. Barbary (Saint Barbara’s church) at the period of the photograph (1860).

This methodology has been implemented using a combination of technologies (OOP/ XML-XSLT / SVG / VRML, see [4]) on various case studies in the city of Krakow (Poland), a UNESCO-listed urban ensemble of great architectural value and diversity. It is important to stress that 3D or 2D representations are calculated at query-time, and not written once for all, each such representation is calculated after having parsed its current “state of information” (bibliography, morphology, etc.).

We describe architectural objects through a hierarchy of classes with the root class factorising the attributes responsible for representing the documentation’s analysis. Each concept isolated detains several blocks of attributes, five mainly qualitative – and nested inside the root class –, one related to the class’ morphology - class specific. Each such concept detains methods relevant for persistence handling in XML files and RDBMS context (mySQL). References on the documentation are stored in another database that describes what the data is (a book, a plan, etc.) and attaches this data to information on what it is about (an edifice, a part of an edifice, etc.). Each object is identified by a unique Id, but its morphology may be described in several XML morphology files if the architectural object evolved through time (which of course is quite common). We propose in line with current technical opportunities a solution in which a unique input (the instance’s XML sheet) has several outputs (i.e. VRML and SVG).



**Fig. 5.** Left, Timeline scene extracts, view on ulica Anny; on left, the timeline slider, in scenes, note the orientation change in Saint Anne church from late gothic to baroque (1500->1700), state of knowledge February 2005.

Right, Four views from the timeline concerning object Ratusz (Town Hall), state of knowledge December 2004; from top left to bottom right dates shown 1316, 1500, 1725, 1850.

### 3 Final Observations

*Informative Modelling’s* goal indeed relates more to J. Bertin’s [2] view (“...a graphic is never an end in itself: it is a moment in, the process of decision making...”)



than to trendy so-called realistic reconstruction. But in the same time the objects displayed inside the graphics we produce “*represent visually a physical thing*” [6] and therefore match this Spence’s definition of scientific visualisation. Informative modelling ultimately intersects questions raised in geometric modelling and in information visualisation. Our research at this stage underlines premises of this methodology:

- Build a theoretical model exploiting what we know about architectural morphologies a priori<sup>1</sup>, a vital point as soon as one wants to allow reuses and comparisons.
- Relate objects to a given architectural scale (from urban layout to simple objects).
- Produce representations dynamically as a reading of the information set’s content in order to display *a picture at time T* of our knowledge. “Dynamic” means here produce graphics at query time, and not produce graphics in which things *move*.
- Develop a vocabulary of graphical signs and codes that can be used to mark objects in order to visualise the above mentioned uncertainty evaluation.
- Produce representations that can be queried in order to retrieve the elements of information that justify the inferences made for the reconstruction.
- Provide a description framework for the information sets inside which each source can be autonomously attached to architectural objects.
- Choose an abstraction level for the representations that matches the message the representation delivers and/or the object’s level of completeness.
- Last but not least, if the graphics does not produce insight on the information sets behind the artefact’s shape, consider it worthless.

The *Informative modelling* methodology, implemented in real-case studies (over 800 edifices or evolutions of edifices, a bibliography of 178 items) does prove at least workable. Our experience also shows that a price has to be paid - faithfulness to the specificity of the architectural heritage, and notably to uncertainty handling.

## References

1. Kantner, J.: Realism vs Reality: creating virtual reconstructions of prehistoric architecture, in Virtual reality in archaeology. Oxford: Archeopress (2000)
2. Bertin, J.: Sémiologie graphique. EHESS (1967) 1998
3. Alkhoven P.: The changing image of the city. A study of the transformation of the townscape using CAD and visualisation techniques. PHD University of Utrecht (1993)
4. Dudek I., Blaise J.Y.: Exploiting the architectural heritage’s documentation. I-Know 03, Graz, Austria (2003) 128-134
5. Tufte, E.R.: Visual Explanations. Graphics Press/ Cheshire (1997)
6. Spence, R.: Information visualisation. Addison Wesley ACM Press (2001)
7. Jurisica, I., Mylopoulos, J. and Yu, E., Ontologies for knowledge management: an information system perspective. Knowledge and Information systems (2004) 380-401

---

<sup>1</sup> i.e the set of fundamental types on which architectural artefacts are based : arches, lintels, etc.



# Computer-Assisted Artistic Pattern Drawing

Hun Im and Jong Weon Lee

Sejong University, School of Computer Engineering,  
Seoul, Korea  
{terehun@naver.com, jwlee@sejong.ac.kr}

**Abstract.** Computer-assisted animation production tools have an important role in animation production industries. They could reduce the production cost and time. In this paper, a new computer-assisted artistic pattern drawing system is proposed. The system automatically adds artistic patterns to 2D animations. The artistic patterns are captured from key frames drawn by an animator and are applied to line drawings. The result shows that the presented approach can copy artistic patterns from key frames and reduce time and cost for producing artistic animations.

## 1 Introduction

A 2D animation production is a labor-intensive task. Master animators draw key frames, and animators draw in-betweens from the key frames to generate smooth motions. In-betweens are frames between two key frames, and the number of in-betweens required for each animation is large. Drawing in-betweens manually is a tedious and time-consuming process.

Many researchers have been developed computer-assisted techniques to automate 2D animation production procedures using computer graphics and image analysis technologies. Levoy[1] and Stern[2] tried to support the entire animation production procedure using vector inputs or raster graphics respectively. In-betweens are automatically generated using key frames and interpolation methods [3][4][5], and automated coloring techniques are developed by matching lines or regions [6][7]. In spite of these efforts, 2D animation procedures have not been yet fully automated, especially for animations that require artistic expressions.

Recently, artistic expressions (e.g., Asian traditional painting styles) are applied to create more expressive animations. The artistic expressions provide an audience more friendly and expressive animations. However, animations with artistic expressions are more difficult to produce than non-artistic animations. Animators have to add artistic expressions manually, but there are few animators that could create animations with artistic expressions. It is also difficult to use automated procedures to add artistic expressions to animations. Researchers developed automated systems to apply artistic patterns to line drawings [8][9]. Those automated systems simply applied the predefined artistic patterns to line drawings, so they cannot present individual artistic styles of animators. Each animator has his/her own style, and that those automated procedures do not easily represent animator's artistic style.

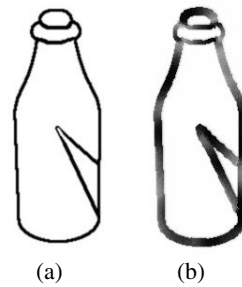
In this paper, a Computer-Assisted Artistic Pattern Drawing (CA<sup>2</sup>PD) system is presented. The CA<sup>2</sup>PD system automates the procedure that adds artistic patterns to line drawings. The CA<sup>2</sup>PD system captures artistic patterns from key frames drawn by an animator and adds the captured patterns to line drawings, so the system can automatically provide artistic styles of animators to 2D animations.

Following this section, the CA<sup>2</sup>PD system is presented in section 2. In section 3, the way to use two key frames to generate in-betweens with artistic expressions of an animator are introduced, and the conclusion is given in section 4.

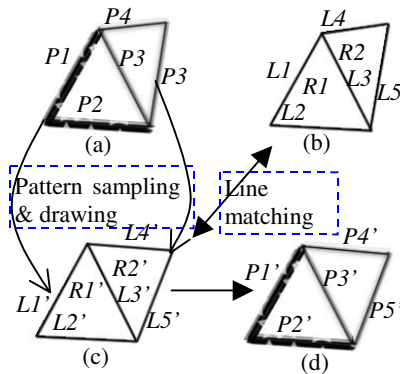
## 2 Computer-Assisted Artistic Patten Drawing (CA<sup>2</sup>PD) System

An animator draws line drawings and adds artistic patterns to the line drawings based on key frames to create animations. The CA<sup>2</sup>PD system mimics the procedure that an animator follows to create animations. The CA<sup>2</sup>PD system captures artistic patterns from key frames and applies those patterns to in-betweens.

Inputs to the CA<sup>2</sup>PD system are key frame sets and target line drawings. The key frame set contains a key frame and a line drawing, which is called a key frame sketch in this paper. An animator adds patterns to a key frame sketch, Fig. 1 (a), to create the key frame, Fig. 1 (b), so the size, the location, and the orientation of the key frame are exactly matched with those of the corresponding key frame sketch. The target line drawings are in-betweens of two key frames, and the CA<sup>2</sup>PD system will add patterns to them using patterns on key frames.



**Fig. 1.** Key frame set  
(a) Key frame sketch (b) Key frame



**Fig. 2.** The procedure of drawing patterns to a target line drawing (a) Key frame (b) Key frame sketch (c) Target line drawing (d) Result drawing

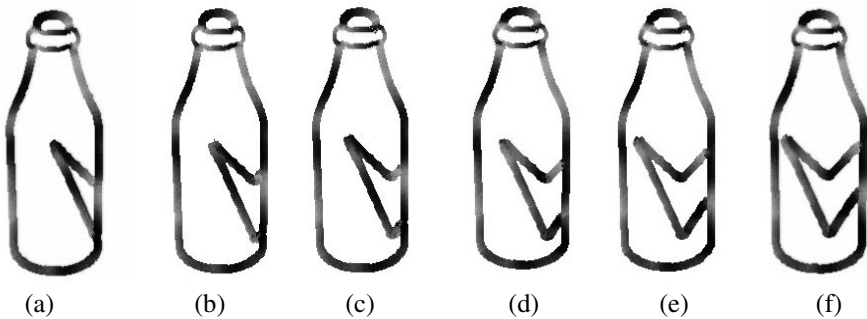
Fig. 2 describes the procedure of adding patterns to a target line drawing. The procedure contains two components, a matching procedure and a pattern drawing procedure. In the matching procedure, regions on the target line drawing are compared with regions on the key frame sketch to detect corresponding regions. From these region correspondences, corresponding line segments between the key frame sketch and the target line drawing are detected. Examples are shown in Fig. 2.  $R1'$  and  $R2'$  on the target line drawing are corresponding regions of  $R1$  and  $R2$  on the key frame sketch.  $L1'$ ,  $L2'$ ,  $L3'$ ,  $L4'$  and  $L5'$  on the target line drawing are corresponding lines of  $L1$ ,  $L2$ ,  $L3$ ,  $L4$  and  $L5$  on the key frame sketch.

After detecting corresponding line segments, the pattern drawing procedure captures patterns on the key frame,  $P1$ ,  $P2$ ,  $P3$ ,  $P4$  and  $P5$ , and draws those patterns on the target line drawing. The result drawing is shown in Fig. 2.

### 3 Creating an Animation Using the CA<sup>2</sup>PD System

In this section, we demonstrate how the CA<sup>2</sup>PD system adds artistic patterns of key frames to in-betweens. Two key frame sets are provided to the system, and the system generates in-betweens that have artistic patterns on two key frames. Two key frames are called a head key frame and a tail key frame as shown in Fig. 3. For line segments on each target line drawing, the CA<sup>2</sup>PD system compares them with line segments on both key frame sketches and captures the artistic patterns, which will be applied to target line drawings using the procedures described in the section 2. Because two key frame sets are used, each line segment on the target line drawing can use a pattern on one of two key frames that is more closely related to that line segment. For example, a line segment on the target line drawing may not detect a corresponding line segment on the head key frame sketch but may detect on the tail key frame sketch.

Fig. 3 demonstrates the quality of the results created by the CA<sup>2</sup>PD system. The CA<sup>2</sup>PD system draws artistic patterns on four in-betweens of two key frames shown in Fig. 3. The frames are created by rotating a bottle along the up direction. As shown in Fig. 3, artistic patterns on two key frames are successfully added to all in-betweens, and the artistic patterns on two key frames are transferred to in-betweens correctly.



**Fig. 3.** In-betweens created by two key frames (a) Head key frame (b)(c)(d)(e) In-betweens (f) Tail key frame

## 4 Conclusion

In this paper, we present a new computer-assisted drawing method, the CA<sup>2</sup>PD system. The CA<sup>2</sup>PD system can apply artistic patterns drawn by an animator automatically on in-between line drawings by capturing artistic patterns from key frames. As a result, artistic expression of an animator can be easily transferred to line drawings. This is the main contribution of the paper. The CA<sup>2</sup>PD system opens a door to adding artistic expressions easily to 2D animations that is not easily done with existing computer-assisted systems even if the system requires improvements.

Currently, the system is only applicable to limited line drawings, which can be divided into enclosed regions. The system could not draw artistic patterns on line segments that are not parts of the enclosed regions, because the system cannot detect corresponding line segments on the key frame sketches. The system also fails for new line segments that are not shown in key frames. These are two important problems that require immediate improvements among many possible improvements.

## References

1. Levoy, M.: A color animation system based on the multi-plane technique, *ACM Computer Graphics* 11 (1977) 65-71
2. Stern, G.: SoftCel – an application of raster scan graphics to conventional cel animation, *ACM Computer Graphics* 13 (1979) 284-288
3. Burtnyk, N. and Wein, M.: Computer-generated key frame animation, *J. Soc Motion Pict Television Eng* 80 (1971) 149-153
4. Burtnyk, N. and Wein, M.: Interactive Skeleton Techniques for Enhancing Motion Dynamics in Key Frame Animation, *Communications of the ACM*, 19 (1976)
5. Lu, J., Seah, H. S. and Tian, F.: Computer-Assisted Cel Animation: Post-processing After Inbetweening, *Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia* (2003)
6. Seah, H. S. and Feng, T.: Computer-assisted coloring by matching line drawings, *The Visual Computer* (2000)
7. Qiu, J., Seah, H. S., Tian, F., Chen, Q. and Melikhov, K.: Computer-Assisted Auto Coloring by Region Matching, *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications* (2003)
8. Hsu, S. C. and Lee, Irene H. H.: Drawing and Animation Using Skeletal Strokes, *Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (1994)
9. Hsu, S. C., Lee, I.H.H. and Wiseman, N.E.: SKELETAL STROKES, *Proceedings of the 6th annual ACM symposium on User interface software and technology* (1993)

# VR-Mirror: A Virtual Reality System for Mental Practice in Post-Stroke Rehabilitation

Jose A. Lozano<sup>1</sup>, Javier Montesa<sup>1</sup>, Mari C. Juan<sup>1</sup>, Mariano Alcañiz<sup>1</sup>, Beatriz Rey<sup>1</sup>,  
Jose Gil<sup>1</sup>, Jose M. Martinez<sup>1</sup>, Andrea Gaggioli<sup>2</sup>, and Francesca Morganti<sup>2</sup>

<sup>1</sup> Medical Image Computing Laboratory (MedICLab),  
Polytechnic University of Valencia (UPV), Camino de Vera,  
46022 Valencia, Spain  
jlozano@dsic.upv.es  
<http://www.ci2b.upv.es/>

<sup>2</sup> Applied Technology for Neuro-Psychology Lab, Istituto Auxologico Italiano,  
Via Peliza da Volpedo 41, Italy, Milan  
[andrea.gaggioli@auxologico.it](mailto:andrea.gaggioli@auxologico.it)

**Abstract.** Developments in basic neurological research and techniques used to train professional athletes suggest that one way of facilitating this learning process of motor schemas is through the use of motor imagery, a training technique in which the procedure required to perform a task is mentally rehearsed in absence of actual physical movement. Clinical studies have shown that rehabilitation of hemiplegic and hemiparetic patients can be improved by integrating physical and mental practice. In this paper, we describe an advanced virtual reality workbench, the VR- Mirror, that it has been designed to support stroke patients with upper-limb hemiplegia in generating motor images. The development of this prototype has been supported by the Commission of the European Communities (CEC) - IST programme (Project I-Learning, IST 2001-38861).

## 1 Introduction

Hemiplegia is total paralysis of the arm, leg, and trunk on one side of the body. The most common cause is stroke, which occurs when a rupture or blood clot reduces blood flow to a specific area of the brain, killing cells and disrupting the abilities or functions they control. Hemiplegia can be caused by damage to a variety of structures including primary sensorimotor areas, supplementary motor, premotor, and parietal cortices, basal ganglia and/or the thalamus [1].

Traditional rehabilitation after stroke focuses on passive (non-specific) movement or on compensatory training of the non-paretic arm. However, only a small percentage (5–52%) of patients regains functional recovery of the upper extremity [2]. This observation may be related to the limited effectiveness of current therapy techniques used for improving upper limb function, and the very small percentage of the patient's day actually spent in upper limb intervention. Clinical research has demonstrated that intensive, repetitive practice of active functional tasks shows more positive outcomes for upper limb rehabilitation.

Mental practice, also called symbolic rehearsal or motor rehearsal, is a training technique in which the procedure required to perform a task is mentally rehearsed in absence of actual physical movement [3]. That is, the subjects repeatedly “rehearse” a motor act in working memory, without producing any overt motor output [4], [5].

Several studies in sport psychology have shown that mental practice can be effective in optimizing the execution of movements in athletes and help novice learners in the incremental acquisition of new skilled behaviours [6]. These studies have generally shown that volunteers who train mentally on a specific task usually display less improvement than those who train physically, although mental practice leads to superior increases in performance compared with a no-practice condition [7]. On the other hand, in recent years several clinical studies have confirmed the contribution of mental practice for improving upper limb motor function after stroke [8], [9], [10], [11]. Even recent neuroimaging experiments using positron emission tomography (PET) and functional magnetic resonance imaging (fMRI) have shown that imagining a motor act is a cognitive task that engages parts of the executive motor system: in particular the supplementary motor area, premotor area, superior parietal lobule, and cerebellum [12]. However, for many patients with damage to the central nervous system, mental simulation of movements can be difficult, even when the relevant neural circuitry has been spared [13].

Another very important aspect to motor skill training applications is the perspective used by the participant. The internal imagery perspective provides an image from the perspective of an active participant, from within the body looking out. Kinesthetic imagery brings in additional senses of touch and the feeling of movement, essential elements in developing images to improve hand-eye coordination. Conversely, visual imagery is generally associated exclusively with the sense of sight. The bulk of evidence supports the use of the internal and kinesthetic perspectives. A number of investigators found that the most successful athletes and experimental participants used the more inclusive kinesthetic perspective even when asked to use an external perspective [3]. This suggests that mental practice maybe more effective in training motor skills when the subject is required to form mental images of the movement to be performed from an internal, egocentric perspective.

Starting from these premises and keeping in mind the recent developments in virtual reality (VR), we have designed, developed and tested a virtual-reality workbench in order to help post-stroke hemiplegic patients evoking motor images and to acquire motor abilities. This workbench has been called VR-Mirror. The technical design and development of this prototype has been carried out by MedICLab (Medical Image Computing Laboratory - UPV (Universidad Politécnica de Valencia – Spain)) and it has adjusted to the features of the clinical protocol developed by the Istituto Auxologico Italiano, which it has also carried out the clinical tests. MedICLab and the Istituto Auxologico Italiano have been partners of I-Learning Project, a European project supported by the Commission of the European Communities (CEC), in particular by the IST programme (Project I-Learning, IST 2001-38861).

## 2 VR-Mirror: Design and Development

In order to obtain the goal pursued in the I-Learning project it was very important to know its clinical rehabilitation strategy before carrying out the design and development of the advanced virtual reality solution.

### 2.1 Clinical Rehabilitation Strategy

The goal of I-Learning rehabilitation system was to help hemiplegic / hemiparetic patients to create compelling motor images that support recovery of motor functions. That is, to evoke powerful imaginative responses using visual cues that draw the patient's attention to the underlying dynamic structure of a movement. This was done following four consecutive steps:

1. Firstly, the motor exercise will be performed by the patient with the healthy limb. This will allow the system to acquire the basic dynamic features of the movement by using a tracking device.
2. Then, the display will depict the mirror image of the movement performed by the unimpaired extremity, which will have been acquired by the system. The observation of the reflected limb will provide a direct perceptual cue of the impaired limb completing a smooth and controlled movement. According to the I-learning hypothesis, this will support the patient in generating a compelling mental image of the movement using an ego-centric perspective (kinesthetic image).
3. The patient will be instructed to mentally rehearse the exercise depicted by the screen.
4. After watching the reflected limb on the screen, the patient will be invited to perform the movement with the impaired extremity by following the mirror image. During the execution of the motor task, the impaired extremity will be tracked and the system will compare in real time the degree of deviation of the actual movement (impaired extremity) from the correct one (healthy extremity) and will give the patient, also in real time, an audio feedback about the correctness of his performance.

The training focuses on well-known motor exercises for the upper extremities and more specifically in pronation / supination forearm and fine fingers movements.

### 2.2 VR-Mirror Hardware

Following this, the goal of the VR-Mirror was to display to the patient a computer 3D representation of their movements with impaired and healthy extremities, using an ego-centric frame of reference. The idea of this workbench can be considered an evolution of the works of Ramachandran [14], who used a mirror box apparatus for the rehabilitation of phantom paralysis and Stevens et al. [15], who developed a motor imagery program for stroke patients making use of a mirror box apparatus.

The relationship between our approach and these works is the goal pursued and the difference is the use of advanced technology to obtain it. In Ramachandran and Stevens approaches was used an apparatus in which mirror images of movement by a

healthy limb (created by a physical mirror) were effectively used to stimulate motor imagery of the movement by the unhealthy limb. Virtual reality technology used in VR-Mirror stimulates motor imagery with computerized mirror image. On the other hand it is possible to capture the patient limb movement and to compare it in an objective way with the movement of the computerized mirror image used as stimuli.

- In *design phase* it was defined the prototype: a horizontal interactive workbench constituted by these components:

- 1 horizontal retro projected screen.
- 1 LCD projector with 1500 lumens at least.
- 1 mirror in order to reflect the projector beam onto the horizontal retro projected screen.
- 1 tracker system with 2 sensors (receivers) in order to track and to capture the hand and the forearm movements of the patient.
- 2 Sets of 5 buttons (one button per finger) placed on both sides of the screen and plugged to the computer through a USB port in order to capture the finger movements of the hands of the patient.
- 2 gloves prepared with belkro in order to place the 2 sensors (receivers).
- 1 high-end graphic capabilities computer with these minimum requirements:
  - Pentium IV 2.6 GHz.
  - Ati Radeon 9800 128Mb.
  - RAM 512Mb. – 1024Mb.
- 1 height adjustable chair in order to fit system dimensions to any user.

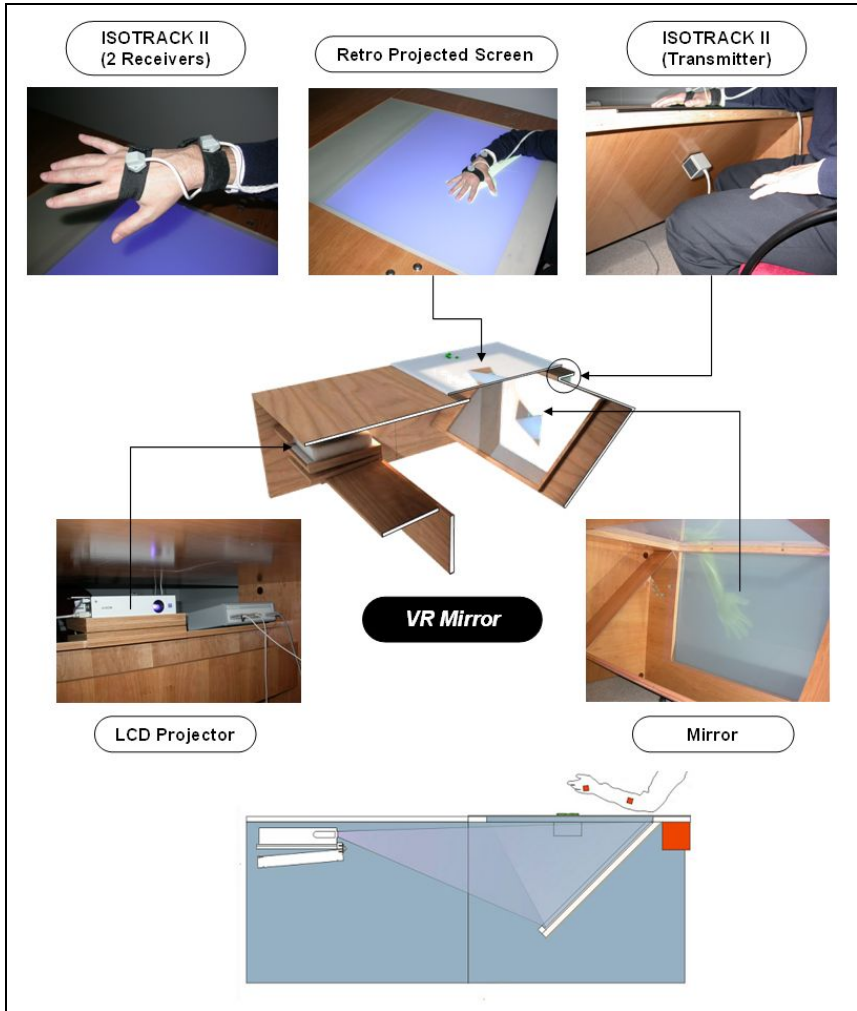
- In *development phase* it was made the prototype. The obtained result is shown in Fig. 1:

The VR-Mirror has been designed as a table that it allow to the patient to seat down in a adjustable chair in order to visualize the virtual environment in a comfortable way. The LCD Projector projects the virtual environment on the inclined mirror (45°). The virtual environment is reflected on the horizontal retro projected screen (translucent metacrilate) and the patient can display it. In order to give to VR-Mirror a solid aspect it has been made in wood.

The control unit of tracker system (Polhemus IsoTrack II) is placed next to the projector (see Fig. 1) because it is the most suitable place. The 2 receivers are placed in the hand and the forearm of the patient with 2 belkro gloves. To both sides of the horizontal retro-projected screen has been placed a set of 5 buttons in order to capture the finger movements of the patient.

In this way the patient visualize the therapeutic virtual environment (VE), a 3D virtual hand and forearm simulating several rehabilitation tasks to be performed by him/her. Moreover, the VE also provide to the patient and an audiovisual feedback in real-time related to the performance of its movements. In this way the patient knows if he or she is performing the rehabilitation tasks correctly. On the other hand, the therapist visualizes the interface used to control the VR-Mirror application.. To obtain this configuration it is necessary to use the two VGA outputs of the graphic card of the PC. One of them is plugged to the LCD Projector and the other one is plugged to PC monitor.





**Fig. 1.** VR-Mirror components: Projector, Mirror, Retro Projected Screen and Tracker System

On the other hand, in order to use the system correctly it is necessary to follow these steps: *Projection Screen Registration* and *Virtual Hand Scale*. These steps must be made by the therapist using the interface of the VR-Mirror application mentioned previously.

#### 1. *Projection Screen Registration:*

In order to register the location (position and orientation) of the projection screen it is necessary to place one receiver of the tracker system in each one of its corners following this order: top-left, top-right, bottom-right and bottom-left.

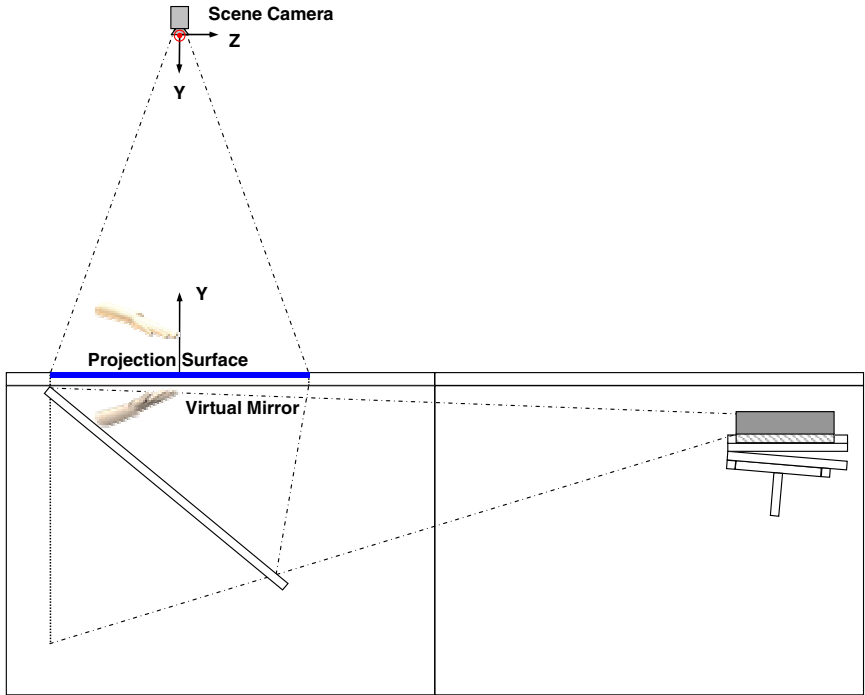


Fig. 2. Projection Screen Registration

In this way also is registered the location of the virtual scene camera as if a real camera was focusing it from above. The location of the projection surface and the scene camera places a virtual mirror in the scene in the same location of the projection surface. Using the virtual mirror it is obtained and showed the reflection of the real hand.

2. *Virtual Hand Scale:*

In order to adjust the virtual hand and the patient's hand sizes it is necessary to introduce the distance size in centimetres from thumb to little finger (see Fig. 3).

2.3 VR-Mirror Software

Regarding the clinical specifications mentioned previously it was designed the VR-Mirror application showed in the Fig. 4.

Every time the therapist uses the VR-Mirror application he/she will create a *session* for the patient. In every session there will be several rehabilitation exercises named *tasks* that the patient will be able to perform.

The first time the patient performs any task it will be necessary to perform it with his/her healthy limb. In order to perform it correctly the patient will follow a rhythmic



Fig. 3. Virtual Hand Scale

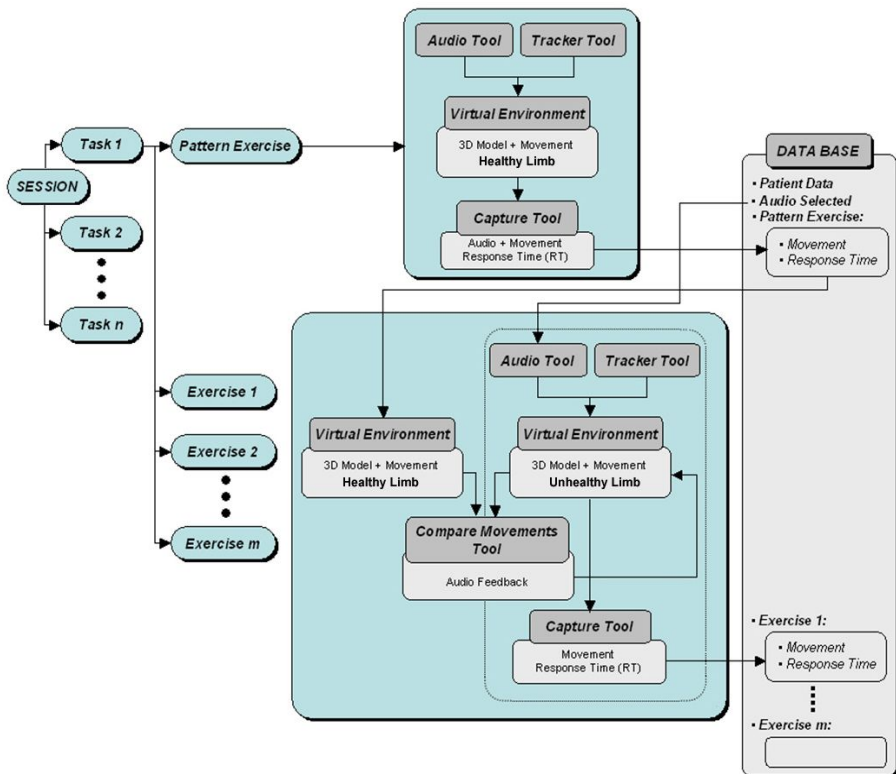


Fig. 4. The design of VR-Mirror application

audio cue provided by the application. On the other hand, the application will capture the movements of the patient. This capture will be used to move the 3D virtual representation of their limb showed in the VE. In this way, the patient will have a visual feedback about the achievement of the task in real time. Finally, the application will

save in a data base the rhythmic audio cue used, the captured movement and the time used to perform it. This first performance will be named *pattern exercise*.

After patten exercise, the patient will try to perform the same task with the unhealthy limb so many times as he/she need. Every attempt will be named *exercise*. In order to help him/her to perform the task correctly the application will show him/her in the VE the pattern exercise saved in the data base. Every time the patient tries to perform the task the application will capture their movements and will show him/her a 3D virtual representation of their unhealthy limb. Moreover, the application will compare both movements and will transfer the patient an audio-visual feedback in order to help him/her to correct their performance in real time. Finally, the application will save in a data base the captured movement and the time used to perform it.

In order to carry out this design it was necessary to define the main tools that should be programmed. Firstly, an interface that allows to the therapist to control the VR-Mirror application in a very easy and intuitive way. A VE in which a realistic 3D upper limb of a person in movement it is showed to the patient. An *Audio Tool* in order to select one of the rhythmic audio cues include by the therapist in the application. A *Tracker Tool* in order to capture the locations of two sensors (receivers) of the tracker system and to transfer them to the 3D upper limb of the VE. A *Capture Tool* in order to capture and to save in a data base the movement of the patient and the rhythmic audio cues used to perform it. Finally, a *Compare Movement Tool* in order to compare in real-time two movements. One of them provided by the *Capture Tool* and the other obtained of the data base.

- In order to program the VR-Mirror application it was used the following software tools and programming languages:

- eStudio:

eStudio is an interactive real-time 3D graphic software used in the field of virtual sets and on-air graphics for TV. eStudio is an open software that allows to develop plug-ins for the easy integration of devices. On the other hand, eStudio allows to create a VE using its authoring tool or programming it with Python. This second option has been the option selected in order to program the interface of the VR-Mirror application, the VE and the tools mentioned previously.

- Visual C++:

Visual C++ has been used in order to program the plug-ins that they will allow communicate the VE with the devices included on the VR-Mirror.

### 3 Results

At present, the VR-Mirror clinical evaluation is carried out in the Physical Rehabilitation Unit of Padua Teaching Hospital from Padua, Italy. It has been tested with a 46-year-old man with stable motor deficit of the upper right limb following stroke. To be exact, the patient has been evaluated for level of impairment and disability. The intervention has consisted of one daily session, three day a week, for four consecutive weeks. This time frame has been chosen because according to available literature the majority of upper extremity rehabilitation programs last 4 to 6 weeks and because

substantial improvement has been demonstrated after 4 weeks of imagery practice [16]. Each therapeutic session has included ½ hour of traditional physiotherapy plus ½ hour of VR-Mirror treatment.

From a clinical point of view the results obtained to date are very interesting. Measurements of wrist function have revealed increases in range of motion during the first phase of intervention, with no losses in movement range occurring after the intervention has been completed. Moreover, the grip strength of the affected right limb of the patient has increased appreciably the patient. On the other hand the patient has been highly motivated throughout the intervention period and has been present at all sessions. From the third week on, he has reported an increase in self-confidence using the affected limb.

From a technical point of view these are the therapist's and the patient's opinions related with the interactive process. From the therapist point of view the VR-Mirror is not portable enough because it is very heavy and difficult to assemble. This is a very important drawback when it is necessary to move it to another room or clinical. The VR-Mirror application is very complete and its use is easy and intuitive. However the therapist thinks it would be very useful to obtain some graphs about the movements of the patient such as speed and times of response, etc. In spite of these drawbacks the therapist opinion is very positive because the clinical results obtained to date are very interesting and optimistic.

From the patient point of view it is uncomfortable to use wire sensors because it troubles the performance of the movements and the change of the sensors from a limb to another one when it is necessary to do it. Also it is uncomfortable every set of 5 buttons used in order to capture the finger movements of the hands of the patient because the distance between them is fixed and sometimes it is not equal to the distance between the fingers of the patient. In spite of these drawbacks the VR-Mirror has succeeded in motivating the patient to carry out the rehabilitation tasks thanks to the ability of the virtual reality to stimulate him/her.

Keeping this in mind, a first conclusion is obtained: from the clinical point of view the goal pursued in the beginning of the project has been obtained with the VR-Mirror. However, in order to improve its clinical efficacy it will be necessary to improve its design.

## 4 Conclusions

From a clinical point of view, VR-Mirror offers an effective combination of motor imagery stimulation (mental practice) and physical performance of rehabilitation tasks (physical practice) that it leads to functional gains in the paretic upper limb of the patients with stable motor deficit of the upper right limb following stroke.

From a technical point of view, VR-Mirror offers very interesting advantages over other approaches used for similar goals:

- The VR-Mirror captures and stores the dynamic features of the hand and the forearm patient movements (healthy and unhealthy limb) and it compares them in an objective way.

- The VR-Mirror provides to the patient the visualization of their movements in an ego-centric perspective that it helps him/her to generate kinesthetic motor images. The kinesthetic or internal perspective brings in additional senses of touch and the feeling of movement, essential elements in developing images to improve hand-eye coordination.
- The VR-Mirror provides to the patient and audiovisual feedback in real-time related to the performance of its movement. In this way he/she knows if it is necessary to rectify it.
- On the other hand, the VR-Mirror provides to the therapist a continuous computerized feedback about the therapeutic process of the patient and their evolution: the rate of the motor skills learning of the patient, etc. In this way the therapist knows if it will be necessary to modify their therapy program [17], [18].

## References

1. Johnson, S. H., G. Sprehn, et al. (2002) Intact motor imagery in chronic upper limb hemiplegics: evidence for activity-independent action representations. *J Cogn Neurosci* 14(6): 841-52
2. Whittall, J., S. McCombe Waller, et al. (2000). Repetitive bilateral arm training with rhythmic auditory cueing improves motor function in chronic hemiparetic stroke. *Stroke* 31(10): 2390-5
3. Driskell, J. E., Copper, C., Moran, A. (1994). "Does Mental Practice Enhance Performance?". *Journal of Applied Psychology*, 79(4), 481-492.
4. Decety, J. and M. Jeannerod (1995) Mentally simulated movements in virtual reality: does Fitts's law hold in motor imagery? *Behav Brain Res* 72(1-2): 127-34
5. Annett J. (1995). Motor imagery: perception or action? *Neuropsychologia*. 33:1395-417
6. Paivio, A. (1985). "Cognitive and motivational functions of imagery in human performance". *Can J Appl Sport Sci*, 10(4), 22S-28S
7. Feltz, D. L., Landers, D.M. (1983). "The effects of mental practice on motor skill learning and performance: a article". *Journal of Sport Psychology*, 5, 25-27
8. Crosbie JH, McDonough SM, Gilmore DH, Wiggam MI. (2004) The adjunctive role of mental practice in the rehabilitation of the upper limb after hemiplegic stroke: a pilot study. *Clin Rehabil*. Feb;18(1):60-8
9. Page, S. J., P. Levine, et al. (2001) A randomized efficacy and feasibility study of imagery in acute stroke. *Clin Rehabil* 15(3): 233-40
10. Stevens, J. A. and M. E. Stoykov (2003) Using motor imagery in the rehabilitation of hemiparesis. *Arch Phys Med Rehabil* 84(7): 1090-2
11. Johnson, S.H. (2004) Stimulation through simulation? Motor imagery and functional reorganization in hemiplegic stroke patients. *Brain & Cognition*. 55(2):328-31
12. Jackson, P. L., M. F. Lafleur, et al. (2001) Potential role of mental practice using motor imagery in neurologic rehabilitation. *Arch Phys Med Rehabil* 82(8): 1133-41
13. Goldenberg, G (1989) The Ability of Patients with Brain Damage to Generate Mental Visual Images. *Brain* 112: 305-325
14. Ramachandran, V. S., & Rogers-Ramachandran, D. (1996). "Synaesthesia in phantom limbs induced with mirrors". *Proc R Soc Lond B Biol Sci*, 263(1369), 377-386
15. Stevens, J. A., & Stoykov, M. E. (2003). "Using motor imagery in the rehabilitation of hemiparesis". *Arch Phys Med Rehabil*, 84(7), 1090-1092

16. Lacourse MG, Turner JA, Randolph-Orr E, Schandler SL, Cohen MJ. Cerebral and cerebellar sensorimotor plasticity following motor imagery-based mental practice of a sequential movement. *J Rehabil Res Dev*. 2004 Jul;41(4):505-24.
17. Meneghini, A. (1997). "Realtà Virtuale ed emiplegia. Report preliminare". Paper presented at the Atti del Convegno Triveneto: "Argomenti di Riabilitazione", Venezia
18. Meneghini, A. (2001). "Motor rehabilitation of the hemiplegic patient through electronic system arranged biofeedback electromiography and virtual reality". Paper presented at the International Council of Psychologists, King Alfred College, Winchester

# Picturing Causality – The Serendipitous Semiotics of Causal Graphs

Eric Neufeld and Sonje Kristtorn

Department of Computer Science, University of Saskatchewan,  
Saskatoon, Sk, Canada S7K5A9  
{eric, wlf203}@cs.usask.ca

**Abstract.** Bayesian nets (BNs) appeared in the 1980s as a solution to computational and representational problems encountered in knowledge representation of uncertain information. Shortly afterwards, BNs became an important part of the AI mainstream. During the 1990s, a lively discussion emerged regarding the causal semantics of Bayesian nets, challenging almost a century of statistical orthodoxy regarding inference of causal relations from observational data, and many refer to BNs now as causal graphs. However, the discussion of causal graphs as a data visualization tool has been limited. We argue here that causal graphs together with their causal semantics for *seeing* and *setting*, have the potential to be as powerful and generic a data visualization tool as line graphs or pie charts.

## 1 Introduction

Causal graphs have been studied as such for more than a decade. Originally introduced as Bayesian nets [3], they demonstrated the computational practicality of purely probabilistic reasoning to a skeptical AI community that considered probability theory was “epistemologically inadequate” even for mundane knowledge. In domains where any variable had relatively few variables directly affecting it, causal graphs offered both a compact representation of joint distributions of many variables, and sound and efficient inference algorithms.

At that time, the ‘causal’ aspect of causal graphs was informal, although abundantly obvious in simple settings such as diagnostic bipartite causal graphs (e.g., the work of Peng and Reggia [7] translated to causal graphs). When Pearl and Verma [5] offered probabilistic definitions of potential and genuine causality that not only gave philosophical justification for this phenomenon, but also suggested an algorithm for recovering causal structure from sufficiently rich numeric data, causal graphs evolved from abstract data structure to philosophical framework.

During the course of building an animated visualization of Gaussian causal graphs, we found that causal graphs also provide a fascinating and semiotically rich visualization of multivariate data [2]. With modest animation, a causal graph can instantly provide a wealth of information to the user, which can be interpreted at many different levels. A naïve user, interested in quickly understanding the structure of a set of



data, can interpret the structure provisionally as causality. A more precise notion would be *causality as elimination of confounding bias*. Less controversially, the graphs can also be interpreted as independence maps. Regardless, the visualization, offers the viewer important insights into the data that may trigger other discoveries.

To illustrate the potential visualization power of causal graphs, the next section presents a visualization of causality and correlation since the early 20<sup>th</sup> century. All of the images are based on screen captures from our visualization tool. (A few minor artistic changes were made manually.) Although all variables are Gaussian, and have been normalized, this is not a limitation of the software.

## 2 Picturing Correlation and Causality

### 2.1 Karl Pearson and Correlation

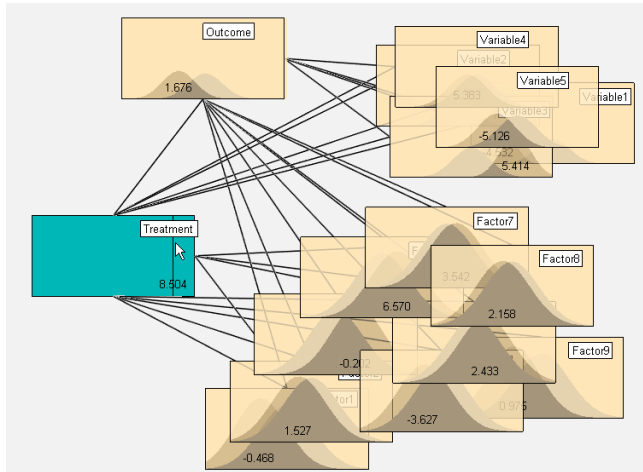
We begin with Karl Pearson, a pioneer of statistics, who strongly opposed any notion of causality as “metaphysical nonsense” [9]. Pearson saw cause and effect as an old and simplistic concept, extreme cases on a continuum of dependence, rendered obsolete by the new ideas of association. He wrote [6]:

The newer and I think truer, view of the universe is that all existences are associated with a corresponding variation among the existences in a second class. Science has to measure the degree of stringency, or looseness of these comcomitant variations, Absolute independence is the conceptual limit at one end to the looseness of the link, absolute dependence is the conceptual limit at the other to the stringency of the link. The old view of cause and effect tried to subsume the universe under these two conceptual limits to experience – and it could only fail; things are not in our experience either independent or causative. All classes of phenomena are linked together, and the problem in each case is how close is the degree of association.

His idea that things are not causative is difficult to refute philosophically. However, generalizing causality to correlation, yields something not only mathematically and conceptually simpler (as when some authors treat probability as a generalization of logic), but also practical, in the sense correlation is much easier to measure than causality.

Let’s visualize Pearson’s world of correlation. Suppose we wish to understand the relationship between some *Treatment*, and an *Outcome* in a world without independence or causality. In that case, all factors are related to the both variables of interest, and so a visualization will look like Figure 1.

In this world, we can pull the treatment and outcome variables to one side, but they cannot be disentangled from the other *Factors*, each of which is correlated with the treatment and the outcome. Another aspect of the complexity, not visible in Figure 1, is that all *Factors* are correlated. As well, Figure 1 shows relatively few other factors. In Pearson’s world, everything is correlated, so every known variable should, in theory, appear in the visualization as a factor. Indeed, it may be presumptuous to imagine that the number of *Factors* is finite or even countable.



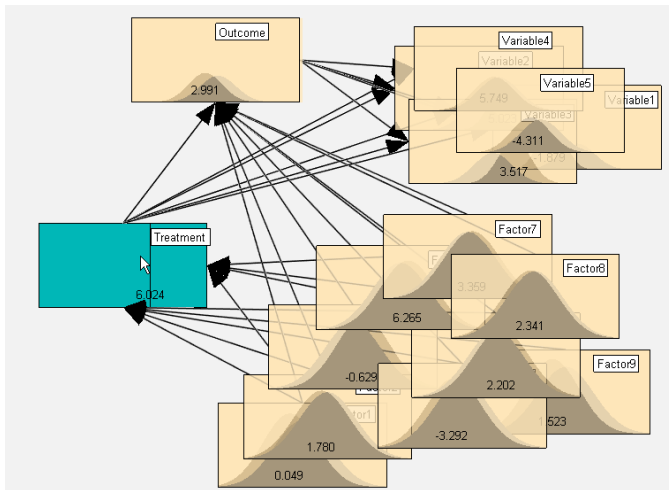
**Fig. 1.** A visualization of Pearson’s world

## 2.2 Fisher and the Randomized Experiment

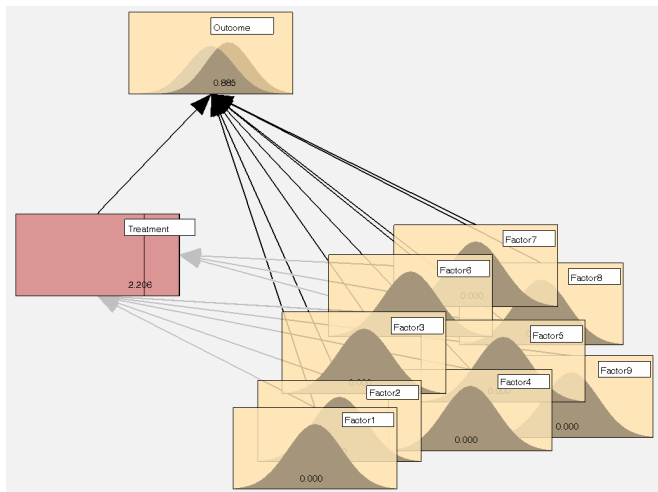
Causality may be philosophically and mathematically troublesome, but it is a major aspect of the human enterprise. Courts of all types attribute cause (blame) to individuals. Physicians determine causes (diseases) of symptoms, and offer prescriptions that will cause the malady to go away. Universities seek causes for declining and shifting enrolments, and so on. Because it seems so fundamental and necessary, for now, we will represent the relationship of causality with an arrow directed from cause to effect. We will sidestep possibly circular causal relationships by only considering instantaneous causal snapshots of the world.

Figure 2 shows a world similar to Pearson’s, except that the edges of Figure 1 are directed. The blue color of the *Treatment* node indicates that this variable has been observed to be at a certain level. The ghosted bell curves show the values of the other variables (*Outcome* and all the *Factors*) prior to the observation and the dark bell curves show the posteriori distributions.

At this point, we have established a direction of causality, dragged our variables of interest to one side, we have selected a subset of the treatment group and observed the a posteriori distributions. The software let us drag the observed value of *Treatment* back and forth, and immediately observe that *Outcome* increases with *Treatment*. But we still are unable to make any quantitative inference about cause and effect, because we are unable to disentangle the confounding effects of the *Factors*. (We are unable to even determine whether the *sign* of causal influence is positive or negative.) This is routine in introductory statistics when arguing against making inferences from observational data, i.e., putting a new treatment on the shelves, letting people “help themselves” and measuring the outcomes. It could be that the treatment is very expensive. People that can afford the treatment may enjoy a better lifestyle in other ways that results in a positive outcome even if the treatment is worsening the outcome.

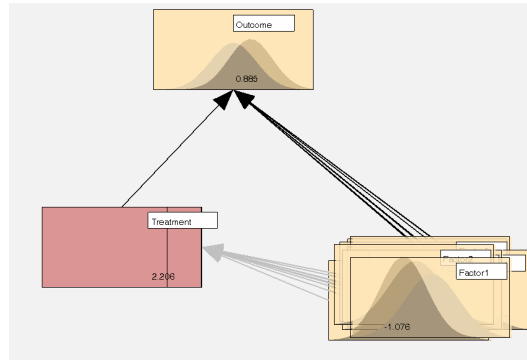


**Fig. 2.** A world with causality



**Fig. 3.** Causal relationships in a scientific experiment

The “genius” [9] of Fisher was the design of the controlled/randomized experiment. To measure the causative effect of a treatment, select a random group of subjects, and administer the treatment in dosages unknown to the subjects. By artificially seizing complete control of administration of the treatment so that the values are unbeknownst to the subjects, the experimenter “erases” the possible causal effect of other factors. Figure 3 visualizes the disentangling of confounds from a causal relationship of interest using the experimental method.



**Fig. 4.** Compact visualization of causality in the experimental setting

The red coloring indicates that the *Treatment* is being manipulated by the experimenters. The greying of arcs from *Factors* to *Treatment* indicates removal of their causal impact on *Treatment*, since the factors cannot affect the dosage administered in the experiment, however strong their force in the world, although they continue to affect outcomes. As well, the causal graph also shows that, in the experimental setting, the *Treatment* (manipulation) becomes *independent* of all the *Factors*. The *Variables* are not shown because they do not affect *Outcome*. The graph shows *Outcome* responds positively to the treatment.

The world in Figure 3 is independent of the number or character of *Factors*, so we redraw it as in Figure 4. This visualization suggests that for the purpose of calculation, we can treat the combination of factors as a single factor. Using discrete probabilities to simplify notation, we can compute the causal effect of *Treatment* (as opposed to  $\sim$ *Treatment* (non-treatment)) from the distribution of outcomes among the experiment subjects as follows as  $P(\text{Outcome}|\text{Treatment}) - P(\text{Outcome}|\sim\text{Treatment})$ . For Gaussian variables, just regress experimental values of *Outcome* against *Treatment*. This will give policymakers a useful prediction of the net effect of the treatment in a population.

The experimental method assumes that the experimental setting does not change the structure of causal relationships in the world. Despite this assumption, there is wide agreement regarding the validity of causal inferences from well-designed experiences. However, certain treatments are unethical or impractical to test, and there can be problems with noncompliance, which returns us to the problem of making inferences from observational data. However, Fisher did not believe it possible to infer causal relationships from observational (i.e., empirical) data.

### 2.3 Pearl and SGS and Causal Graphs from Observational Data

The introduction briefly introduced causal graphs, originally called Bayes nets. For most early students of Bayes nets, the primary importance of Bayes nets was as an abstract data structure permitting efficient representation and inference. We follow Pearl's graph-theoretic and probabilistic notation and terminology [8].

Practitioners observed informally that arcs in causal graphs were typically oriented in the direction of causality. Pearl and Verma [5] provided a philosophical justification for this phenomenon that generalized it to the entire graph structure.

**Definition. (Potential Cause)** (Pearl and Verma)  $A$  is a potential cause of  $C$  if there is a variable  $B$  and a context (set of variables)  $S$  such that

- i.  $A, B$  are independent given  $S$ ,
- ii. there is no set  $D$  such that  $A, C$  are conditionally independent given  $D$ , and
- iii.  $B, C$  are dependent.

To understand this definition, suppose that  $A, B, C$  satisfy the relationship of potential cause above when  $S$  is the empty context. Figure 5 shows all possible orientations of arcs connecting  $A, B, C$  as shown in Figure 5.

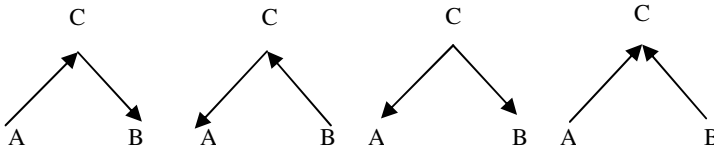
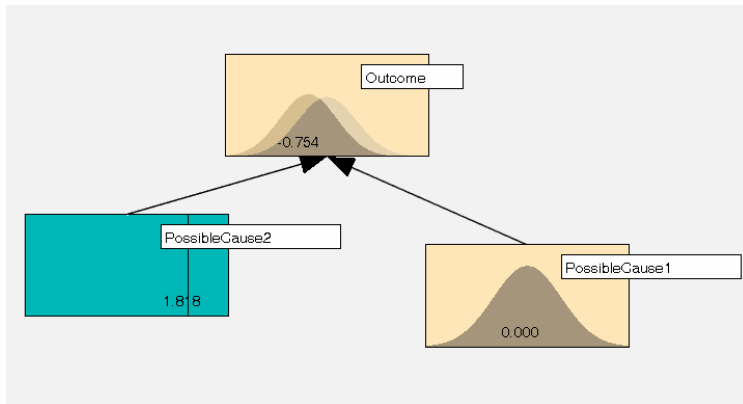


Fig. 5.

The head-to-head graph is the only graph consistent with the independencies in the above definition. The first two graphs, where  $B$  is causally downstream effect from  $A$ , or vice-versa, would imply a dependence between  $A$  and  $B$ . Similarly, if  $A, B$  were produced by a common cause  $C$ , one would expect an unconditional dependence between  $A$  and  $B$  as common effects.

From the perspective of information visualization, it is serendipitous that so simple a structure as a node-link diagram would be consistent with such a careful definition of causality. Node-link diagrams themselves bring semiotic power. Ware [15] points out that node-link representations, such as those appearing in these interfaces, form an important class of perceptual graphical codes: a circular line, though just a mark on paper or a monitor, is interpreted as an object at some level in the visual system (or, in our case, as a variable). Similarly, linking lines imply relationships between objects. Arrows are also interesting: Tversky [13] mentions that Horn [1] catalogues 250 meanings for arrows, and suggests they are useful in particular for representing causality. Thus such a simple representation for causal relationships between variables also appears to have powerful cognitive/semiotic validity.

In the diagrams here, direct manipulation of variables is achieved by sliding values back and forth. This produces instantaneous changes in linked variables that indicate quantitative probabilistic and causal relationships. This is consistent with our most immediate impressions of causality, which in everyday life are associated with the direct manipulation of objects. When we manually move an object in our environment, we correctly perceive that we cause its displacement. This feeling of direct control is considered to be a central principle of contemporary direct manipulation interfaces[15]. We move a computer mouse and through a complex train of events, a



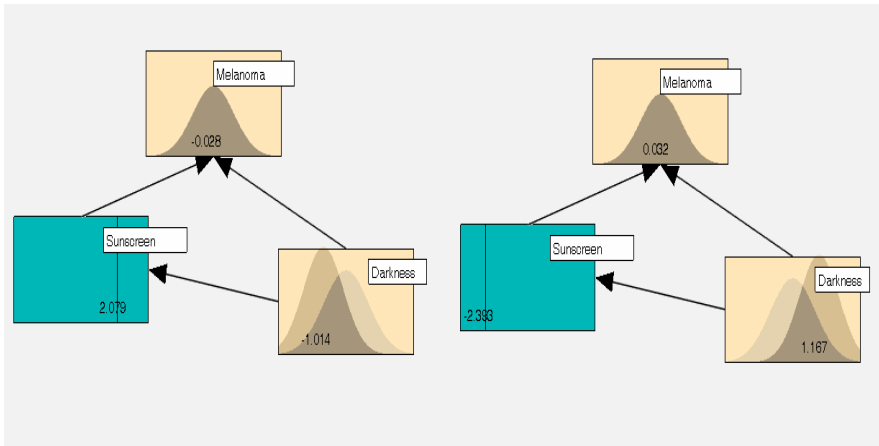
**Fig. 6.** Finding the structure of an experiment in Nature

cursor or other object moves on the screen or a window is resized. Because of the immediacy of the visual change corresponding to the hand movement, we feel that we have directly manipulated something even though there is nothing direct about it. The key requirements for this illusion of direct manipulation are rapid visual feedback ( $< 100$  msec.) and a natural visual relationship between the user's action and the on-screen event, e.g., we move our hand left, the object moves left. Though the optimal correspondence is by no means known, the result of successful direct manipulation is a feeling of transparency such that "the user is able to apply intellect directly to the task; the tool itself disappears" [8]. It is not true that any onscreen change concomitant with a user action is perceived as direct manipulation and the circumstances under which the illusion is achieved are likely to be complex. But certainly object movement linked to hand movement is most likely to promote the illusion.

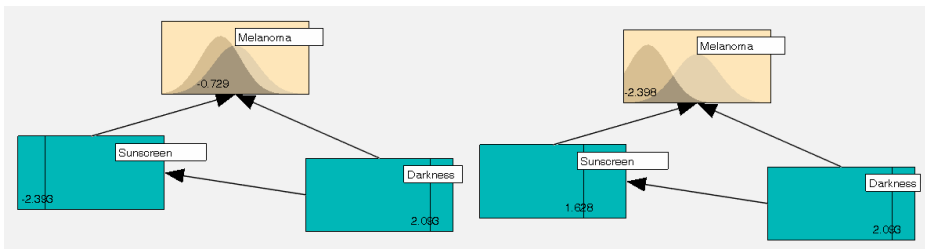
Figure 6 depicts a simple causal graph consistent with the probabilistic assumptions of the Pearl-Verma definition, which tells us how to discover the structure of a scientific experiment in nature. Notice the strong similarities of the graph in Figure 6 to the depiction of the scientific experiment in Figure 4. When discovering causality in observational data, by definition, we can't apply a control. Thus the possible causes must a priori obey the independence relationships we artificially create in the experimental setting.

We now turn to an inference problem we have discussed before [2], which we call the *Sunscreen* problem. Suppose that observational data suggests that *Melanoma* is probabilistically barely affected by *Sunscreen* usage. On the one hand, policy-makers are concerned because holes in the ozone layer suggest a need for health authorities to advocate the use of products that shield people from the possibly harmful effects of the sun. On the other hand, the data has made them suspicious that some of the sunscreens being marketed contain carcinogens.

Figure 7 shows that *Melanoma* doesn't respond much to *Sunscreen*, in terms of condition expectation based on the observational data. However, there happens to be data available on *Darkness*, a measure of darkness of skin. The expectation of



**Fig. 7.** Melanoma does not respond much to Sunscreen



**Fig. 8.** Manipulating *Sunscreen* after fixing *Darkness*

*Darkness* varies inversely with *Sunscreen*. That is, people with darker skin use less sun screen. Perhaps darkness is acting as a confound and masking the effect of sun screen. A data analyst might consider fixing the value of *Darkness* at some level, and then manipulating the *Sunscreen* variable. Figure 8 shows the result.

For dark-skinned people, incidence of *Melanoma* decreases as *Sunscreen* increases. (The same relationship obtains when skin type is fixed at a lower value, although the mean shifts to the right.) *Sunscreen* now results in decreased *Melanoma* for all skin types, although the mean shifts with skin type because of the natural protection offered by pigmentation. Given this set of results, a policymaker can make an educated conjecture regarding the result of enforced/encouraged sunscreen usage.

Finding these relationships requires a certain amount of experience with data analysis. It would be useful if such discoveries could be found by less experience users. That is, we would like to calculate a probability called  $p(M|set(S))$ , where we use the *set* functor to distinguish *manipulating* of the variable (by enforcement) from

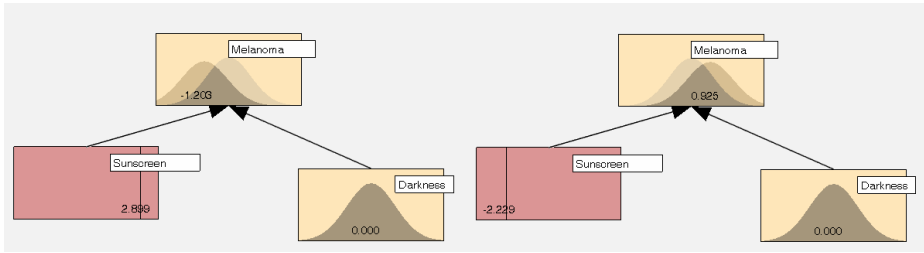


Fig. 9. Resulting of enforcing *Sunscreen* usage

seeing (as calculated from the observational data). People expect the probability of a manipulation to be the same as the probability of observation, namely, the conditional probability  $p(M|S)$ . In a discrete formulation,  $D$  indicates dark skin and  $\neg D$  indicates light skin. The target probability, using the ordinary axioms of probability, is

$$P(M|\text{set}(S)) = p(M|S) = p(M|DS)p(D|S) + p(M|\neg DS)p(\neg D|S).$$

But this formulation is wrong, because although  $P(M|DS)$  correctly counts the incidence of melanoma among dark-skinned sunscreen users, multiplying it by  $p(D|S)$  will not predict the posterior incidence of melanoma among dark-skinned sunscreen users in the new scenario *after the manipulation*, since it does not count dark-skinned persons currently not using sunscreen, who will use it in the new scenario. Thus,  $p(M|DS)$  should be multiplied by the proportion of *all* persons who are dark-skinned. A symmetric argument applies to the second multiplicative term, and thus, the complete correct formulation is

$$P(M|\text{set}(S)) = p(M|DS)p(D) + p(M|\neg DS)p(\neg D).$$

Pearl shows that in terms of the graphical structure, this is equivalent to erasing the arrow from *Darkness* to *Sunscreen* and recomputing the conditional probabilities accordingly, and points out that the theoretical basis for this goes back to Stotz and Wold [12]. Figure 9 shows the result of using our graphic software to depict this.

Visually, enforcing *Sunscreen* usage (that is, the effects of manipulating *Sunscreen*) have *exactly* the same structure as Figure 4, which depicts the manipulation of a treatment variable in the experimental setting. That is, the *setting* calculation is like doing a “thought” “controlled experiment” on the data. Of interest is that Sloman and Lagnado [10] have found that the notions of *seeing* and *setting* also have cognitive validity, yet another confirmation of the information content of these simple diagrams.

### 3 Discussion and Conclusions

These simple visualizations show that there is much in common with the notion of inferring causality from observational data as presented by Pearl (and also Spirtes *et al* [11]) and the notion of causality in scientific experiments. We consider this testi-



mony to the expressive power of the causal graph formalism as a generic statistical visualization tool. The novice can put in observational data, add causal arrows, and trivially compute the consequences of policy decisions. A more seasoned data analyst can interpret the results in Figure 9 more precisely as a calculation of removing confounding bias, subject to the causal assumptions implied by the arrows. At the very least, the structure allows the user to efficiently explore dependencies.

The visualizations can also be used to convey other ideas. Some statisticians and philosophers have argued strongly against the usefulness of causal graphs for causal inference, by asking, in effect, “given a world as complicated as Figure 2, what are the chances we will find three variables, of interest, and with exactly the probabilistic relationships depicted in Figure 6?” The visualization doesn’t answer this question, but does demonstrate that it is a very good question. As well, the visualizations are useful depictions of mathematical ideals. (Experiments themselves are ideals in some sense.)

The issue of causal inference from observational data continues. Regardless, as Ware [15] writes, “seeing a pattern can often lead to a key insight, and this is the most compelling reason for visualization”. Even if the causal semantics of causal graphs are philosophically troublesome, the visual structure has the potential to become a useful generic data representation for many kinds of information.

The software has remained simple as regards graphics. The only animation capability presently is that of responding immediately to changes in levels of variables, made possible by the implementation of an efficient algorithm for manipulation of Gaussians. Otherwise, we have intentionally left the presentation sparse, partly by way of minimizing what Tufte calls “chart junk” [13], but partly to maximize accessibility to this tool. Members of our group are exploring ways of improving the expressive power of the graphs using animations that exploit the human cognition system’s ability to perceive causation.

## References

1. Horn, R.E. *Visual Language*. MacroVu, Inc Bainbridge Island, WA: (1998).
2. Neufeld, E, Kristtorn, S., Guan, Q., M Sanscartier, and Ware, C. Exploring Causal Influences. In *Proceedings of Visualization and Data Analysis 2005*, San Jose, 52-62
3. Pearl, J. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA (1988)
4. Pearl, J. (2000) *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge
5. Pearl, J and Verma, T. A theory of inferred causation. In *Principles of Knowledge Representation and Reasoning: Proceedings of the 2<sup>nd</sup> International Conference*, San Mateo (1991) 441-452
6. Pearson, K. *The Grammar of Science*, third edition. London, Adam and Charles Black (1911)
7. Peng, Y and Reggia, J.A. Plausibility of diagnostic hypotheses. In *Proceedings of the 5<sup>th</sup> National Conference on AI*, Philadelphia, (1986) 140-145
8. Rutowsky, C. (1982) An Introduction to the Human Applications Standard Interface, Part 1: Theory and Principles. *BYTE* 7(11): 29-310

9. Shipley, B. 2000. *Cause and Correlation in Biology: A User's Guide to Path Analysis, Structural Equations and Causal Inference*. Cambridge University Press: Cambridge UK
10. Sloman, Steven A., and Lagnado, David A. Do We "do"? *Cognitive Science* **9** (2005) 5-39
11. Spirtes, P., Glymour, C., and Scheines, R. *Causation, Prediction and Search*, Springer-Verlag, New York (1993)
12. Stotz, R.H. and Wold, H.O.A (1960) Recursive versus nonrecursive systems: An attempt at synthesis. *Econometrica* 28, 417-427
13. Tufte, E.R. *The Visual Display of Quantitative Information*. Graphics Press (1983)
14. Tversky, B. Zacks, J., Lee, P. and Heiser, J. Lines, Blobs, Crosses and Arrows: Diagrammatic Communication with Schematic Figures. In *Proceedings of the First International Conference on Theory and Application of Diagrams 2000*: 221-230
15. Ware, C. (2000) *Information Visualization: Perception for Design*, Morgan Kaufman, San Francisco

# Xface: Open Source Toolkit for Creating 3D Faces of an Embodied Conversational Agent

Koray Balci<sup>1</sup>

ITC-irst, Trento, Italy

[balci@itc.it](mailto:balci@itc.it)

<http://xface.itc.it>

**Abstract.** Xface, the new version of our open source, platform independent toolkit for developing 3D embodied conversational agents is presented. The toolkit currently incorporates four pieces of software. The core Xface library is for developers who want to embed 3D facial animation to their applications. XfaceEd editor provides an easy to use interface to generate MPEG-4 ready meshes from static 3D models. Xface-Player is a sample application that demonstrates the toolkit in action and XfaceClient is used as the communication controller over network.

## 1 Introduction

Embodied conversational agents (ECAs) have become a popular subject in both research and industry domains such as entertainment industry, customer service applications, human computer interaction and virtual reality. We believe that the lack of free and open source tools for creation and animation of faces limit the further research on the field. With that in mind, we have started an open source initiative which will provide the research community a tool for generating and animating 3D talking agents, namely Xface (see Balci, K. 2004). In this paper, we present the improved version with keyframe animation technique together with an added module for communication over the network.

The toolkit basically relies on MPEG-4 Facial Animation (FA) standard (see Standard 1997) so that it is able to playback standard MPEG-4 Facial Animation Parameters (FAP) streams, though it also supports keyframe animation as an alternative methodology.

Because of the wide audience we aim for, the architecture of Xface is meant to be configurable and easy to extend. All the pieces in the toolkit are operating system independent, and can be compiled with any ANSI C++ standard compliant compiler. For animation, toolkit relies on OpenGL and is optimized enough to achieve satisfactory frame rates (minimum 25 frames per second are required for FAP generating tool) with high polygon count (12000 polygons) using modest hardware.

The toolkit involves four pieces of software as output of the Xface project. Those are the core library, an editor for preparation of faces, and a sample player. In the following subsections, an overview of those are presented.

## 2 Core Library

The core Xface library is responsible for loading the face models and corresponding MPEG-4 information, as well as streaming FAP data and handle deformation and rendering in order to create facial animation.

According to MPEG-4 FA standard, there are 84 feature points (FP) on the head. For each FP to be animated, corresponding vertex on the model, and the indices to the vertices in the zone of influence of this FP should be set. We also define the type of deformation function to be applied to each zone. In the present implementation, deformation is defaulted to a raised cosine function applied to each FP region as shown in Eq. 1, where  $\Delta v_p$  denotes euclidean distance that the point  $p$  in FP zone of influence should be translated in FAP direction, while  $d_p$  is the  $p$ 's and  $d_{max}$  is the farthest point's distance in region to the FP.  $w$  is a weight value defined for every deformation and  $fap$  is the FAP value.

$$\Delta v_p = \left( 1 + \cos \left( \pi \times \frac{d_p}{d_{max}} \right) \right) \times w \times fap \quad (1)$$

Raised cosine achieves satisfactory results, although one can extend the library easily to use different deformation strategies like Radial Basis Functions (see Hagiwara et. al. 2002), Free Form Deformations (see Terzopoulos et. al. 1997).

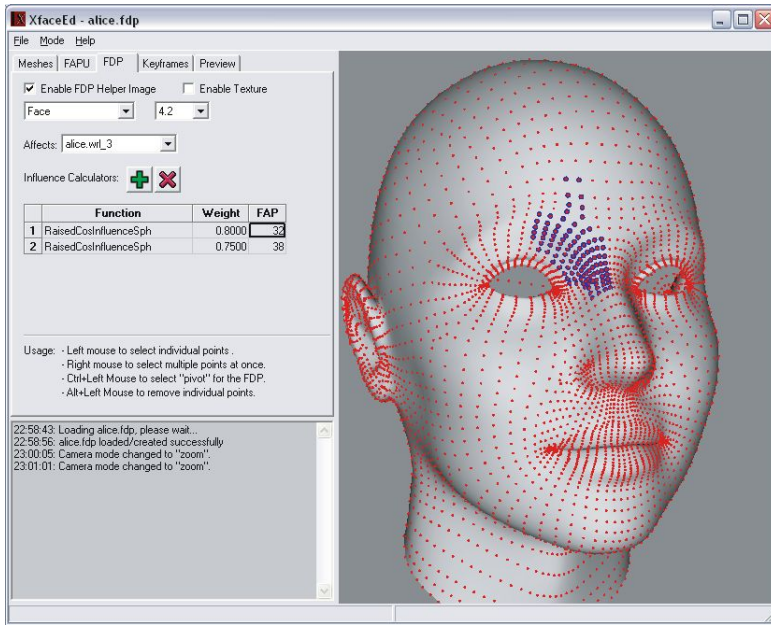
As an alternative method, one can also define keyframes representing visemes and certain emotions and expressions using XfaceEd. Then, instead of using MPEG-4 FAP's, one can create an ECA using simple keyframe interpolation technique by blending keyframes of different categories in real time.

## 3 XfaceEd

MPEG-4 FA represents a standard way to parameterize the animation process, but one has to define certain parameters such as Facial Animation Parameter Units (FAPU) and FPs on the 3D model manually. XfaceEd simplifies this process by providing an easy way for defining the FAPU, FP regions, weights, and parameters for manipulating and animating the static face models. Figure 1 is a screenshot to illustrate the process of setting FP regions. Static 3D models can be created by any 3D content creation package, and imported to XfaceEd. The output is a configuration file in XML format. This helps the definition of deformation rules and parameters externally, so that nothing is hard coded inside Xface library. One can change the 3D face model with little work, without doing any programming. The configuration file is interpreted by the library and the respective deformation rules are generated automatically. For the alternative keyframe animation approach, one can also set the morph targets for visemes, expressions and emotions using XfaceEd as well.

## 4 XfacePlayer

XfacePlayer is a sample application that demonstrates how one can implement a face player using the core library. You can load an MPEG-4 FAP file, an

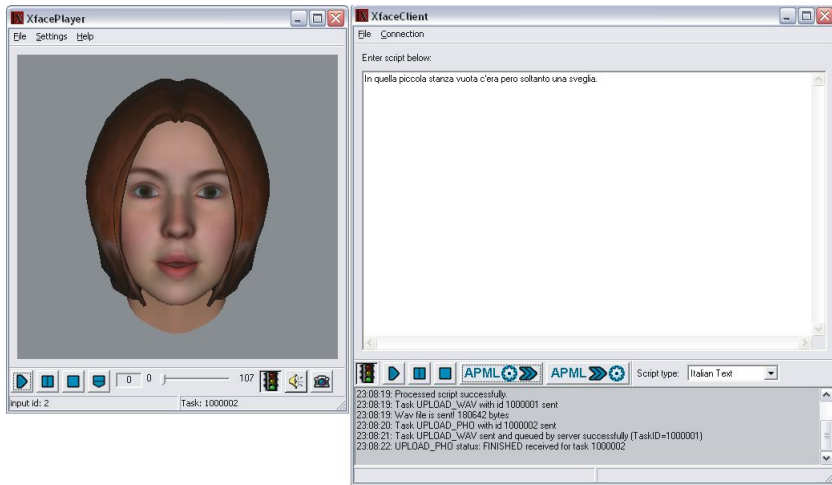


**Fig. 1.** Setting deformation rules using XfaceEd

audio file (for speech) together with the configuration file created by XfaceEd with a couple of function calls and have a basic talking head implementation. Alternatively, keyframe method together with a text to speech (TTS) module, can be used. In addition, the player can be remotely controlled over the network using XfaceClient.

## 5 XfaceClient

XfaceClient is meant to be the communication engine for the ECAs. As input to XfacePlayer, one has to supply audio for speech together with MPEG-4 FAPs or phoneme durations decorated with emotion and expression tags. This can be achieved by using various scripting languages, TTS synthesizers and FAP generators. Xface does not have any means for creation of these input internally but supports various external tools such as Festival (see Taylor et. al. 1998) speech synthesizer, apml2fap FAP creator 1999, APML (see Pelachaud et. al. 2002) and SMIL-AGENT scripting languages. Even some other face player other than XfacePlayer can be plugged. XfaceClient is configured by means of an XML file for supporting these and make external application calls to feed XfacePlayer with the necessary input for facial animation without any hard coding. The communication is done by standard TCP/IP protocol and various tasks and data can be uploaded to the player with an XML based open messaging schema.



**Fig. 2.** XfacePlayer and XfaceClient working together

## 6 Conclusion

In conclusion, Xface, our open source, MPEG-4 based 3D talking head creation toolkit is presented in this paper. Development is still in progress, an early version of the toolkit is available from our website for download.

## References

- DeCarolis, N., Carofiglio, V., Pelachaud, C.: From Discourse Plans to Believable Behavior Generation. Proc. Int. Natural Language Generation Conf. New York (July 2002).
- Taylor, P. A., Black, A., Caley, R.: The Architecture of the Festival Speech Synthesis System. Proc. of 3rd ESCA/COCOSDA Workshop on Speech Synthesis. (1998) 147–152.
- Lavagetto, F., Pockaj, R.: The Facial Animation Engine: Towards a High-Level Interface for Design of MPEG-4 Compliant Animated Faces. IEEE Transaction on Circuits and Systems for Video Technology. **9-2** (1999) 277–289.
- ISO/IEC JTC1/WG11 N1901: Text for CD 14496-1 Systems. Fribourg Meeting, November 1997.
- Kojekine, N., Savchenko V., Senin, M., Hagiwara, I.: Real-time 3D Deformations by Means of Compactly Supported Radial Basis Functions. Proc. of Eurographics02, 35–43. September 2002.
- Faloutsos, P., van de Panne, M., Terzopoulos, D.: Dynamic Free-Form Deformations for Animation Synthesis. IEEE Transactions on Visualization and Computer Graphics. **3-3** (1997) 201–214.
- Balci, K.: Xface: MPEG-4 Based Open Source Toolkit for 3D Facial Animation. Proc. Advance Visual Interfaces. (2004).

# Interactive Augmentation of 3D Statistics Visualizations

Peter Bergdahl

Arts and Communication, Malmö University, 205 06 Malmö, Sweden  
KF019069@stud.mah.se

**Abstract.** Interactive 3D statistics visualizations have certain limitations. Through experiments with a 3D model, I have created a number of interaction techniques – the Reference Grid, Callouts and Interactive Blending – to address and avoid these limitations.

## 1 Description of Visualization Core

The 3D model used for visualizing global statistics is a simple geosphere with control points distributed evenly over the whole surface. These control points are displaced relative to the center of the sphere using a high resolution grayscale displacement map, representing the percentage values of a statistics dataset.

When data is applied to the model, the elevation of the different areas of the globe changes. Areas without data, most commonly oceans, represent either the average or the minimum value of the dataset, and will provide a reference for the other areas. If using an average is appropriate, values above the average of the dataset will proportionally raise a country or control point above the neutral ‘sea’ level, and values below the average will make the points sink into the globe. If an average is not used, all countries will rise accordingly.



**Fig. 1.** As seen in the example, a texture map is applied to the sphere. This texture map is generally a map of the earth with the continents color-coded to separate them from each other, and the borders of all countries lined in black. The map not only helps people to identify continents and countries, it is essential to their understanding of the model.

## 2 Inadequacies and Their Solutions from a Use Perspective

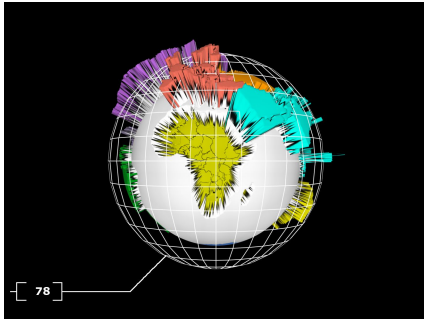
My experiments with the visualization core have demonstrated some inherent problems from a use perspective.

**Comparing Areas.** When statistics are applied to change the elevation of parts of a globe, the viewer naturally has problems to compare how similar in elevation the various parts are. Another problem related to comparison is if the areas that the user is looking at are hidden by other areas or the globe itself.

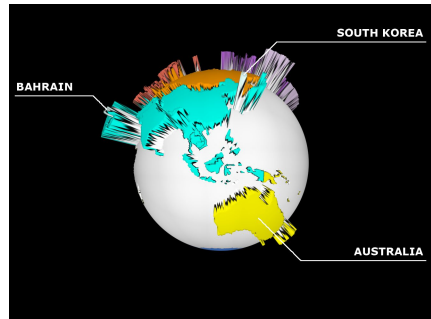
**Orientation and Information.** Depending on the dataset, it can be easy or hard to see similar values in the visualization. Part of the visualization will be hidden, and many countries can have values that are almost the same. There is also a problem with orientation when the viewer does not know exactly where a specific country lies.

**Comparing Datasets.** Sometimes a dataset is divided into several factors, such as percentage of female and male population in employment. There is also the possibility that two datasets make a lot more sense when they are compared to each other.

Here are my solutions to these problems:



**Fig. 2.** The **Reference Grid** can be set to match a specific country, allowing the user to compare the height and value to another country anywhere on the model



**Fig. 3.** **Callouts** can point out countries that have the same values, or be used with a search function, providing the viewer with easier orientation and navigation

**Interactive Blending Between Datasets and Combining Datasets.** By creating two instances of the model using different datasets or parts of a dataset, superimposing them and shifting their transparency, it is possible to show several statistical factors and their relationships and differences.

**Final Comments.** These are naturally only some of the challenges and solutions related to interactive 3D visualizations of global statistics. Further research could be directed towards visualizing excerpts from datasets, image based statistics and adding new layers of related information through new 3D shapes.



# Author Index

- Aish, Robert 151  
Alcañiz, Mariano 241  
Ali, Kamran 115  
André, Elisabeth 1
- Bade, Ragnar 138  
Balci, Koray 263  
Barone, Rossano 77  
Bergdahl, Peter 267  
Biermann, Peter 25  
Blaise, Jean-Yves 230  
Boulanger, Pierre 13  
Braquelaire, Achille 163
- Carpendale, Sheelagh 103, 185  
Cheng, Peter C.-H. 77  
Christie, Marc 40  
Cox, Richard 53
- Denzinger, Jörg 103  
Dorfmueller-Ulhaas, Klaus 1  
Dudek, Iwona 230
- Fujishiro, Issei 175
- Gaggioli, Andrea 241  
Gil, Jose 241  
Götzelmann, Timo 115  
Granier, Xavier 163  
Grawemeyer, Beate 53
- Hartmann, Knut 115  
Heinen, Torsten 222  
Hinrichs, Klaus 218  
Hinrichs, Uta 185  
Hogue, Christopher W.V. 90
- Igarashi, Takeo 175  
Im, Hun 237
- Juan, Mari C. 241
- Kerautret, Bertrand 163  
Kristtorn, Sonje 252
- Latoschik, Marc Erich 25  
Lee, Jong Weon 237  
Lozano, Jose A. 241
- Machap, Rumesh 40  
Martinez, Jose M. 241  
Mason, Kaye 103  
May, Martin 222  
Montesa, Javier 241  
Morganti, Francesca 241  
Mori, Yuki 175  
Müller, Sebastian 127
- Neufeld, Eric 252  
Normand, Jean-Marie 40
- Ohki, Yoshihito 206  
Olivier, Patrick 40
- Pattison, Eric 185  
Pickering, Jonathan 40  
Pintilie, Greg D. 90  
Preim, Bernhard 138
- Rehm, Matthias 1  
Rey, Beatriz 241  
Ritter, Felix 138  
Rodrigues Jr., José F. 65  
Ropinski, Timo 218
- Schmidt, Benno 222  
Schödl, Arno 127  
Scott, Stacey D. 185  
Steinicke, Frank 218  
Strothotte, Thomas 115
- Takahashi, Shigeo 175  
Takeshima, Yuriko 175  
Taylor, Robyn 13  
Terrenghi, Lucia 198  
Torres, Daniel 13  
Traina, Agma J.M. 65  
Traina Jr., Caetano 65  
Tuekam, Brigitte 90
- Wachsmuth, Ipke 25  
Woodbury, Robert 151
- Yamaguchi, Yasushi 206