Lecture Notes in Computer Science 5277

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
Alfred Kobsa
University of California, Irvine, CA, 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
University of California, Los Angeles, CA, USA
Doug Tygar
University of California, Berkeley, CA, USA
Gerhard Weikum
Max-Planck Institute of Computer Science, Saarbruecken, Germany

Arjan Egges Arno Kamphuis
Mark Overmars (Eds.)

Motion in Games

First International Workshop, MIG 2008
Utrecht, The Netherlands, June 14-17, 2008
Revised Papers

Springer

Volume Editors

Arjan Egges
Arno Kamphuis
Mark Overmars
Universiteit Utrecht
Games and Virtual Worlds Group
PO Box 80.089, 3508TB Utrecht, The Netherlands
E-mail: {egges,arnok,markov}@cs.uu.nl

Library of Congress Control Number: 2008939495

CR Subject Classification (1998): H.5, I.6.8, I.2.1, I.2.9-11, I.3-4, J.5

LNCS Sublibrary: SL 6 – Image Processing, Computer Vision, Pattern Recognition, and Graphics

ISSN	0302-9743
ISBN-10	3-540-89219-2 Springer Berlin Heidelberg New York
ISBN-13	978-3-540-89219-9 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

springer.com

© Springer-Verlag Berlin Heidelberg 2008
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12550432 06/3180 5 4 3 2 1 0

Preface

From June 14–17, 2008, the Center for Advanced Gaming and Simulation (AGS), Utrecht University, in collaboration with the NLGD Festival of Games, organized a Workshop on Motion in Games in Utrecht. Motion plays a crucial role in computer games. Characters move around, objects are manipulated or move due to physical constraints, entities are animated, and the camera moves through the scene. Even the motion of the player nowadays is used as input to games. Motion is currently studied in many different areas of research, including graphics and animation, game technology, robotics, simulation, computer vision, and also physics, psychology, and urban studies. The goal of the Motion in Games workshop was to bring together researchers from this variety of fields to present the most recent results and to initiate collaboration.

The MIG 2008 workshop hosted over 30 internationally renowned researchers who all presented their ongoing work on topics such as crowd simulation, motion capture, path planning and facial animation. This volume is a collection of the papers presented during the workshop. Since this volume was published after the workshop, the authors of the papers adapted their content in order to include any discussions that took place during the workshop itself. All final contributions were carefully checked by the workshop organizers.

The Motion in Games workshop was a very successful event that has set the starting point for interdisciplinary collaborations and for novel research ideas following the interesting discussions that took place. We are very happy with the outcome of the workshop and the excellent contributions by the participants, collected in this volume.

August 2008

Arjan Egges
Arno Kamphuis
Mark Overmars

Sponsoring Institutions

This workshop was sponsored by the GATE project[1] and the NLGD Festival of Games[2].

The ambition of the GATE project is to develop an international competitive knowledge base with respect to game technology, and to train the talent required to enhance the productivity and competitive edge of small and medium-sized creative industrial companies. The project will substantially improve the competitiveness of companies producing (tools for) games and simulations by providing direct access to new technology and by technology transfer projects. This will lead to larger companies, encourage the founding of new companies, and attract companies from other countries to The Netherlands. The project will make people aware of the possibilities of gaming in public sectors such as education, health, and safety by performing pilots in these areas. As a result, gaming and simulation will become more commonly applied in these sectors, leading to quality improvements and cost reductions. The GATE project is funded by the Netherlands Organization for Scientific Research (NWO) and the Netherlands ICT Research and Innovation Authority (ICT Regie).

The NLGD Festival of Games is an annual recurring event. The NLGD offers a platform to the strongly upcoming game industry. Too often one tends to think about games as merely games for the youngsters. The NLGD shows by means of activities, publications and events that games and the game industry have a huge social, economic and cultural impact. The worldwide market for games is 30 milliard and the current Dutch market is 0.7 milliard euro in sales consisting of games and simulations (source: NLGD). The Dutch game industry has grown in the last five years by 10% (source: PWC).

[1] http://gate.gameresearch.nl

[2] http://www.nlgd.nl

Table of Contents

Texture Synthesis Based Simulation of Secondary Agents

Marios Kyriakou and Yiorgos Chrysanthou

Department of Computer Science
University of Cyprus
mkyriakou@gmail.com, yiorgos@ucy.ac.cy

Abstract. Realistic simulation of virtual characters is essential in many applications and particularly in computer games. Having a realistic simulation of an agent is a complex matter since there are many different ways in which the real equivalent would have responded in real life. In addition a human viewer is very accustomed to observing other humans so the slightest inconsistency will be picked up. The complexity of the task increases when we address the simulation of groups of virtual agents interacting amongst them.

In this paper we argue that, given the current state of the art, example based methods provide a lot of promise in addressing the realistic simulation. We start by discussing the recently proposed example based methods and then continue to present a novel approach based on texture synthesis. Although it is still work in progress, some preliminary results indicate that this is a viable approach that could potentially achieve results not so easily reached with rule based methods.

1 Introduction

In recent years we have seen a great advancement in computer graphics imagery. Possibly one of the few remaining challenges is the realistic depiction of humans. Virtual humans present difficulties both in terms of rendering and in terms of behavior simulation and animation. This is particularly true in situation where we have a number of humans interacting. Although we have seen a lot of progress in the field recently, and we are gradually seeing more and more virtual humans being included in real-time applications such as games, we are far from having a satisfactory solution. In this paper we concentrate on the multi-agent behavior simulation, and in particular concentrate on simulating the behavior of pedestrian characters.

Some of the most popular approaches used in crowd simulation are either particle based or rule based. The former tend to have less personalization for each agent, while the latter require a lot of carefully crafted rules to get right. Over the last couple of years we have also seen some data-driven techniques appearing in the literature. The latter attempt to create a simulation by stitching together example behaviors observed in real-world video. The data-driven approaches

© Springer-Verlag Berlin Heidelberg 2008

have the advantage that they can capture a lot of variation and subtle behaviors that would require a lot of painstaking rules to encode in a rule based system. In addition they do that without involving the subjective definition of rules by a modeler.

In this paper, we propose a novel data driven approach that is based on the principles of texture synthesis. Our main difference over the other data driven methods is that we do not process pedestrians individually, but we hierarchically synthesize whole areas that may contain several pedestrians inside. This has the possibility of capturing better the interaction between neighboring agents. Moreover, since the existing texture synthesis literature is so rich, there is a large arsenal of techniques that can readily be borrowed in order to solve issues that might arise in our approach.

The method we present here is work in progress. It has been implemented and some initial results have been obtain that seem to support our expectations.

2 Related Work

Our work combines crowd simulation with texture synthesis. The bibliography is very large in both areas, so here we will briefly mention only the most relevant to our work. Research in crowd simulation has been an active theme in a number of fields, such as computer graphics, civil engineering, physics, sociology and robotics. There are various popular methods for simulating crowds. Some derive ideas from fluid mechanics, [1,2], while others make use of particles and social forces [3,4]. However, the most popular approaches are rule-based. The seminal work of Reynolds [5] proposed one of the earliest rule-based simulations, which focused on flocking behaviors for animal "crowds". Various works followed [6,7], both for animal and human crowds.

The works mentioned above focus mainly on the navigation aspect of each individual and the general flow of the crowd. A number of works do look beyond the navigation aspect. Some build a cognitive decision making mechanism for rule definitions. In the work of Terzopoulos et. al. [8] a range of individual traits, such as *hunger* and *fear*, are defined for simulated fish, generating appropriate behaviors. Funge et. al. [9] simulates agents that not only perceive the environment but also learn from it and use domain knowledge to choose the most suitable behavior out of a predefined set. In theory, these methods can be applied to simulated crowds, however they are rather inefficient. In addition they are complicated to define as most rule based systems are, when striving for more realistic behaviors.

Applied to crowd simulation, the work of Musse et. al. [10] takes into account sociological aspects while defining a behavioral model. Sung et. al. [11] represent the set of behaviors as graph (a finite state machine) with probabilities associated to the edges. These probabilities are updated in real-time based on a set of behavior functions. In the work of Farenc et. al. [12] and Thomas et. al. [13], information stored within the environment triggers the agents to perform various actions.

Data driven approaches have also been employed in crowd simulations. Metoyer and Hodgins [14] allow the user to define specific examples of behaviors. Musse et. al. [15] using vision techniques to extract paths from a video for a specific environment. Paris et. al. [16] use motion tracking to extract detailed behaviors from a crowd of people in various small scale environments. In all three cases the examples are used to refine an underlying behavior model.

Brogan and Johnson [17] use the statistics from observed pedestrian paths to improve the navigation model. In the work of Lai et. al. [18] a motion graph approach is used for synthesizing group behavior. Ashida et. al. [19] model subconscious upper body actions with a statistical distribution which is extracted from real video of individuals. This model is used in conjunction with the internal cognitive and psychological state of the agents, in order to produce more realistic pedestrian walking. In the more recent work of Lerner et. al. [20], a database of human trajectories is learned from videos of real crowds. The trajectories are stored along with some representation of the stimuli that affected them. During a simulation an agent extracts from the environment a set of stimuli that possibly effects its trajectory and searches the database for a similar one and copying a trajectory from it.

Texture analysis and synthesis have been around since the 50's [21] in the fields of Psychology, Statistics and Graphic Computers. Using pixel-based texture synthesis, Efros and Leung [22] developed a non-parametric sampling technique, where the composition of textures is done repeating the matching of neighborhood surrounding the processed pixel in the texture that is being composed with the input-texture. Based on this technique, Wei and Levoy [23] developed their own algorithm, using a pyramid of composition, which allows the use and examination of smaller neighborhoods for better and faster results. Simultaneously, they applied tree structured vector quantization for the acceleration of algorithm. An alternative approach was presented by Ashikhmin [24] where the space and time for the search is drastically decreased. Hertzmann et. al. [25], combining the techniques of Wei and Levoy and Ashikhmin in a common frame, achieved enough interesting results and opened new regions for applications.

Patch-based texture synthesis maintain the general structure, creating new textures based on the composition per piece. The algorithm of Efros and Freeman [26] aligns the neighboring limits of certain processed pieces, from an overlap region and then executing a technique of minimum-error-boundary-cut in this overlap region, so as to decrease the imperfections of the overlap. This technique has been adopted in many new algorithms even for 3D composition of textures and from which we took enough elements and for our own algorithm.

3 Overview

The aim of this work is to produce pedestrian simulations based on example data. Our examples come from real-world video footage of people, taken with an overlooking static camera. The captured video is manually analyzed to extract the static geometry and the trajectories of the people. This extracted data can

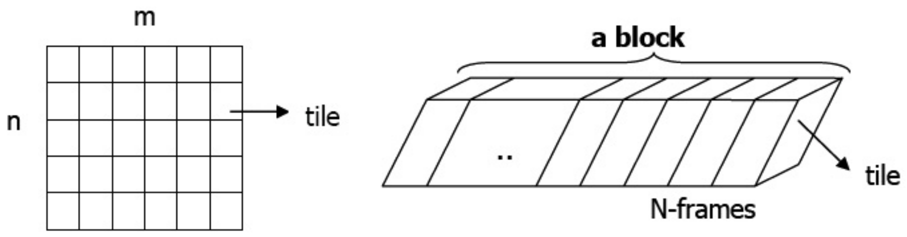


Fig. 1. The frames of the input video are partitioned into tiles (left). The same tile over N consecutive frames is a block (right).

be seen as a simplified video where at each frame we have the colored features - people and static geometry - over a neutral background. This video, or 3D texture, forms the input to our algorithm. Every frame of the input video is segmented into *mxn* square *tiles*. N consecutive frames of the same tile form a *block*, see Figure 1. These blocks are the basic unit on which the algorithm operates.

Our method proceeds in two steps. At preprocessing, the input video is analyzed, and the blocks are placed into a tree structure for easier access. Then at run-time an output 3D texture is created by combining and blending together selected input blocks. The output 3D texture does not need to be the same size as the input. However, both its dimensions needs to be a multiple of the side of the input tiles. The static geometry and the first K frames ($K < N$) need to be pre-defined and are used as the starting point of the algorithm. (See Section 6 for the values of K as well as N that we have used.) The algorithm proceeds in scan-line order using the K frames of a tile as a query into the tree in order to find a good match and bring the "best matching" block of N frames. At the end of an iteration over all the tiles, we have a video extended by $N - K$ frames.

4 Initialization Phase

This is the pre-processing phase in which the database is prepared in an easy to search way. It starts by first creating the 3D blocks and then proceeds to distribute them on the leaves of the *example tree*.

4.1 Creation of 3D Texture Blocks

Having a video with tracked trajectories of individuals we need to construct a large database with 3D blocks that will form the examples which will be used to synthesize new trajectories in the synthesis phase. From the video we extract 3D textures: every frame is split to 2D *mxn* tiles, see Figure 1 left. If we extend these 2D tiles in time we get the 3D block, see Figure 1 right. In order to enrich our database with a larger number of examples we overlap the tiles. The overlap is done by shifting the grid of tiles by a few pixels iteratively in either direction until we get all possible segmentations of the frame.

4.2 Creation of the Example Tree

The 3D blocks created above are placed in the database, and arranged in a tree structure, in order to have faster search in the synthesis phase. The tree has six levels, with the internal nodes used for partitioning the data and only the leaf nodes actually holding the block data. The criteria used for the partitioning at the internal nodes are based on the count of pedestrians at the following locations of a block:

Level 1. Present in the K^{th} frame.
Level 2. Leaving from the west side between the K^{th} and the N^{th} frames.
Level 3. Entering through the west side between the K^{th} and the N^{th} frames.
Level 4. Leaving from the east between the K^{th} and the N^{th} frames.
Level 5. Entering through the east side between the K^{th} and the N^{th} frames.

In order to save memory, the tree actually stores only references to the location of each block, while all the input data is stored just once in a separate common table.

5 Synthesis Phase

In the synthesis phase we start from a given set of trajectories, K frames long, and extend them in time. In our implementation we take as our starting point the trajectories of the last K frames of the input video. As already mentioned, the size of the output frames can actually be different from the input if desired. The synthesis works one block at a time in scan-line order, in the manner of texture synthesis [23]. For each block we first search in the example tree to find the best match and then add it to the output. We will look at these two steps in the following sections and also look at some tuning to the basic algorithm that we have done in order to overcome certain problems we encountered.

5.1 Search for the Best Matching Block

In the spirit of texture synthesis, we look for the best matching 3D block by considering the already constructed neighborhood, both in space and in time. To do that we form a *query* that consists of the N frames of the northern, the western and the north-western 3D block, as well as the K frames of the tile that are already there, see Figure 2.

We examine the query to find the values for the five criteria and use them to traverse to the corresponding leaf of the example-tree. In this way we end-up to a leaf that contains 3D blocks that are similar in these five hard-constrains. In the leaf we evaluate the dissimilarity of the example blocks by comparing their neighborhoods against the query. Firstly, we match each pedestrian from the example with a pedestrian from the query. The couples that are selected are the ones with the smallest dissimilarity value, see Figure 3. The dissimilarity

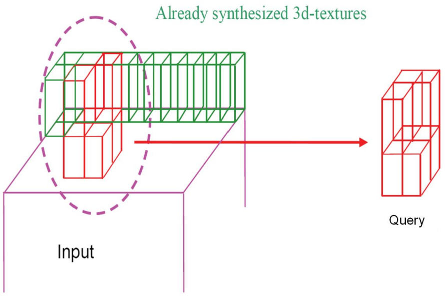


Fig. 2. We form a query using the already constructed neighborhood of the block, and the already existing K frames

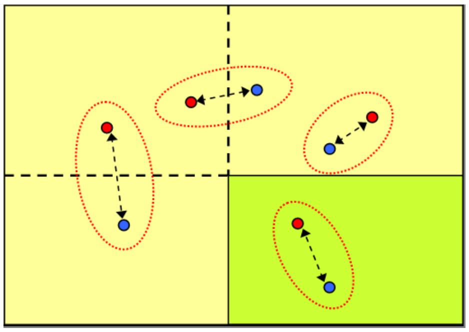


Fig. 3. First we match the pedestrians from the example (blue color) with the pedestrians from the query (red color) and then we measure the distance between the couples

is calculated using the following measurement function which is the sum of the distances between the couples through all N frames.

$A = \sum_{f=1}^{N} \sum_{i=1}^{Nped} \sqrt{((x_{query}^{f,i} - x_{example}^{f,i})^2 + (y_{query}^{f,i} - y_{example}^{f,i})^2)}$

where $Nped$ is the number of pedestrians in the query, f runs over the frames and $x_{query}^{f,i}$ is the x coordinate of pedestrian i in frame f of the query. If we have a pedestrian that is not present in a frame (has left or not entered yet) then we add to A a penalty factor:

A = A + Penalty.

Having found the L most similar 3D blocks, where L is a predefined number, we choose one of them randomly.

5.2 Creation of the New 3D Texture

Once the block is selected, we need to merge it into the output that has already been constructed. Copying the selected 3D block and pasting as is does not give smooth transition between the query and the selected block, see Figure 4. This problem is solved using interpolation between all matched couples (between the

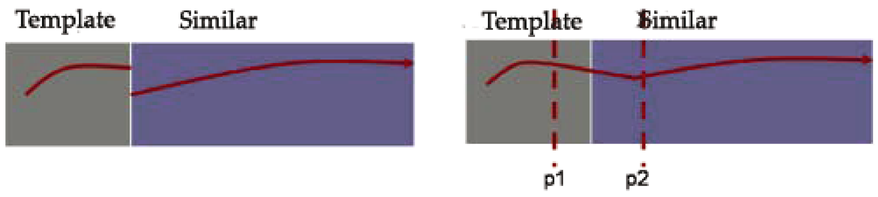


Fig. 4. We may not have a smooth transition between the query and the selected 3D block (left). In order to achieve smooth transition, we can use interpolation (right).

pre-existing and the new 3D block) and creating the new synthesized 3D-texture. We find the two points (frames) where the two trajectories have the less difference between them (P1 from the existing and P2 from the new) and we make the cut on these points. Between these points we create a piece of new trajectory which is the result after the interpolation of the position of the pedestrian at the points P1 and P2, see Figure 4 (right). We do this for every couple and we create the new synthesized 3D block.

The synthesized 3D block with N frames, is inserted in the output, to replace the existing K frames. The new N-K frames that have actually added to the data come from the input data that are real trajectories of real pedestrians. After we apply the algorithm for a complete loop over all the m x n tiles we have extended the trajectories by N-K frames.

5.3 Problems with the Synthesis

Bouncing Characters. In the algorithm as described up to here, if we have a pedestrian moving in a direction opposite to the scanline-order used in the composition, it might create problems. The problem arises from the fact that the query accounts only for the 3 sides that are already in the output and has no way for accounting for people entering from the other side. Anyone coming from the part of the neighborhood that we did not examine was not considered when those textures were processed and synthesized, therefore would have no trajectory entering. It would bounce back.

We solved this problem by considering a circular neighborhood for the first K frames, see Figure 5, and calculating a Neighborhood Similarity Measurement Function for each pedestrian. This is an indication for the presence of pedestrians near the examined 3D block.

We calculate a weight measurement (B) for every pedestrian which is in the radius we examine and in the 3D blocks that we have already processed:

$$B = \sum_{f=1}^{K} \sum_{i=1}^{Nped} \sqrt{((x_{Tcentre} - x_{pedestrian}^{f,i})^2 + (y_{Tcentre} - y_{pedestrian}^{f,i})^2)}$$

where $(x_{Tcentre}$ and $y_{Tcentre}$ are the x and y coordinates of the centre of the tile and $x_{query}^{f,i}$ is the x coordinate of the i^{th} pedestrian in the f frame. In the search process after we end-up at a leaf, we choose a number of 3D blocks that have similar B values. This means that the example that we will finally choose will have similar indication for the presence of pedestrians and in this we are

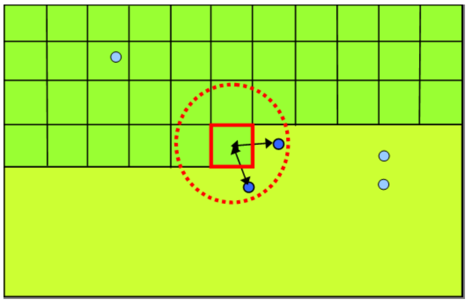


Fig. 5. For every pedestrian which is in the radius we examine we sum his distance from the centre of the tile

considering these incoming pedestrians. For these 3D blocks we calculate the dissimilarity value A to find the examples most similar to the query.

Insufficient Examples in the Database. Since our input data is finite, there is always the possibility that a query defines behaviors substantially different from any of the examples in the database. In that case the dissimilarity values will be very high and choosing any of the examples will give unsatisfactory.

To solve this problem we create another database with smaller blocks than the initial, i.e. we use multiresolution on the size of the tiles. We use 1/2 length x 1/2 width of the initial. Thus, if we cannot find a similar 3D block matching a query, we divide the query to 4 equal parts and for each one of these smaller blocks we do a search in the second database with the smaller examples. The synthesis phase for these smaller blocks is the same.

6 Results

To test our algorithm we created an example database using real data. From the roof of a 5 storey building we used a static camera to capture a video approximately 5 minutes length. Using a semi-automatic system we tracked the pedestrians, and we extracted the position (x,y) of each one of them in every frame.

At consecutive frames the positions of the same pedestrians are most likely to be identical or very close. Thus, we sampled the data every 1:5 and the number of the frames were reduced. In total we had about 10.000 frames. Every frame was divided in tiles by m columns and n rows (m=6, n=5, total 30 tiles). Setting the window size at 420 x 350 points, the tile size was 70 points. Overlapping the tiles (every 14 points in x and y) we have 26 tiles every column and 21 tiles every row (total 546 tiles in a frame). In order to create the 3D blocks we set as N=60 (the number of the consecutive frames for each block) and K=15. So, we created about 5.460.000 3D blocks. A big number of these were actually empty and were discarded. Only about 1.700.000 carried some information (positions of pedestrians) and were stored in the database using the 6-levels tree.

Constructing the output using the 3D blocks of real data means that in effect we are giving to our virtual agents the behaviors that are observed in the real data. If we have pedestrians in the video which are avoiding each other and have natural and plausible behavior, then this will be present in the synthesized output.

Our preliminary results show that indeed it works as expected. Of course this is work in progress, and more fine tuning is needed before we have quality results but the work is promising since the behavior of the input data seems to be maintained in the synthesized frames.

7 Discussion and Future Work

In this paper we present a novel crowd simulation technique based on texture synthesis principles. Since, we do not make any assumptions on the behavior of the pedestrians, we can take examples from any real situation and from those we can synthesize new crowd behavior (new trajectories) that can run indefinitely. As a future work, it would be interesting to have input from several observed situations in order to produce more complex crowd behavior. Thus, we can combine several inputs and can simulate the behavior of the crowd that is shown in real complex situations. In this way, we can simulate the conditions in a real city (crosswalks, shops, entrances, squares, malls, dense and sparse crowds), enriching our database, capturing and analyzing video-clips from these places.

References

1. Hughes, R.L.: The Flow of Human Crowds. Annual Review of Fluid Mechanics 35, 169–182 (2003)
2. Treuille, A., Cooper, S., Popovic, Z.: Continuum crowds. ACM Trans. Graph. 25(3), 1160–1168 (2006)
3. Helbing, D., Molnár, P.: Social force model for pedestrian dynamics. Phys. Rev. E 51(5), 4282–4286 (1995)
4. Heigeas, L., Luciani, A., Thollot, J., Castagné, N.: A physically-based particle model of emergent crowd behaviors. In: Graphicon (2003)
5. Reynolds, C.W.: Flocks, herds, and schools: A distributed behavioral model. Computer Graphics 21(4), 25–34 (1987)
6. Ulicny, B., Thalmann, D.: Towards interactive real-time crowd behavior simulation. Computer Graphics Forum 21(4), 767–775 (2002)
7. Loscos, C., Marchal, D., Meyer, A.: Intuitive crowd behaviour in dense urban environments using local laws. In: TPCG, pp. 122–129 (2003)
8. Terzopoulos, D., Tu, X., Grzeszczuk, R.: Artificial fishes: autonomous locomotion, perception, behavior, and learning in a simulated physical world. Artif. Life 1(4), 327–351 (1994)
9. Funge, J., Tu, X., Terzopoulos, D.: Cognitive modeling: Knowledge, reasoning and planning for intelligent characters. In: Rockwood, A. (ed.) Siggraph 1999, Computer Graphics Proceedings, pp. 29–38. Addison-Wesley Longman, Los Angeles (1999)

10. Musse, S.R., Thalmann, D.: A model of human crowd behavior: Group interrelationship and collision detection analysis. In: Computer Animation and Simulation, pp. 39–52 (1997)
11. Sung, M., Gleicher, M., Chenney, S.: Scalable behaviors for crowd simulation. Comput. Graph. Forum 23(3), 519–528 (2004)
12. Farenc, N., Boulic, R., Thalmann, D.: An informed environment dedicated to the simulation of virtual humans in urban context. Computer Graphics Forum 18(3), 309–318 (1999)
13. Thomas, G., Donikian, S.: Modelling virtual cities dedicated to behavioural animation. In: Eurographics 2000. Blackwell Publishers, Malden (2000)
14. Metoyer, R.A., Hodgins, J.K.: Reactive pedestrian path following from examples. In: CASA 2003: Proceedings of the 16th International Conference on Computer Animation and Social Agents (CASA 2003), Washington, DC, USA, p. 149. IEEE Computer Society, Los Alamitos (2003)
15. Musse, S.R., Jung, C.R., Braun, A., Junior, J.J.: Simulating the motion of virtual agents based on examples. In: ACM/EG Symposium on Computer Animation, Short Papers, Vienna, Austria (2006)
16. Paris, S., Pettre, J., Donikian, S.: Pedestrian Reactive Navigation for Crowd Simulation: a Predictive Approach. Computer Graphics Forum 26(3), 665–674 (2007)
17. Brogan, D., Johnson, N.: Realistic human walking paths. In: 16th International Conference on Computer Animation and Social Agents, 2003, pp. 94–101 (2003)
18. Lai, Y.C., Chenney, S., Fan, S.: Group motion graphs. In: SCA 2005: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 281–290 (2005)
19. Ashida, K., Lee, S., Allbeck, J., Sun, H., Badler, N., Metaxas, D.: Pedestrians: creating agent behaviors through statistical analysisof observation data. In: Computer Animation, 2001. The Fourteenth Conference on Computer Animation. Proceedings, pp. 84–92 (2001)
20. Lerner, A., Chrysanthou, Y., Lischinski, D.: Crowds by Example. Computer Graphics Forum 26(3), 655–664 (2007)
21. Gibson: The Perception of the Visual World (1950)
22. Efros, L.: Texture Synthesis by Non-parametric Sampling, 1033–1038 (2000)
23. Wei, L.: Fast Texture Synthesis Using Tree-Structured Vector Quantization. In: Computer Graphics Proceedings of SIGGRAPH 2000, pp. 355–360 (2000)
24. Ashikimin, M.: Synthesizing Natural Textures. In: Proceedings of ACM Symposium on Interaction 3D Graphics, pp. 217–226 (2001)
25. Hertzmann, A., Jacobs, C.E., Oliver, N., Curless, B., Salesin, D.H.: Image Analogies. In: ACM Computer Graphics (Proceedings of SIGGRAPH 2001), pp. 327–340 (2001)
26. Efros, F.: Image Quilting for texture synthesis and transfer. In: SIGGRAPH 2001 Conference Proceedings, pp. 341–346 (2001)

Using the Corridor Map Method for Path Planning for a Large Number of Characters

Roland Geraerts, Arno Kamphuis, Ioannis Karamouzas, and Mark Overmars

Institute of Information and Computing Sciences, Utrecht University
3508 TA Utrecht, the Netherlands
roland@cs.uu.nl

Abstract. A central problem in games is planning high-quality paths for characters avoiding obstacles in the environment. Current games require a path planner that is fast (to ensure real-time interaction) and flexible (to avoid local hazards). In addition, a path needs to be natural, meaning that the path is smooth, short, keeps some clearance to obstacles, avoids other characters, *etcetera*.

Game worlds are normally populated with a large number of characters. In this paper we show how the recently introduced Corridor Map Method can be extended and used to efficiently compute smooth motions for these characters. We will consider crowds in which the characters wander around, characters have goals, and characters behave as a coherent group.

The approach is very fast. Even in environments with 5000 characters it uses only 40% of the processing power of a single core of a CPU. Also the resulting paths are indeed natural.

1 Introduction

One of the main challenges of applications dealing with virtual environments is path planning for characters. These characters have to traverse from a start to a goal position in the virtual world without colliding with obstacles and other characters. In the past twenty years, many algorithms have been devised to tackle the path planning problem [1,2]. These algorithms were mainly developed in the field of robotics, aiming at creating a path for one or a few robots having many degrees of freedom. Usually, much CPU time was available for computing a *nice path*, which often meant a short path having some clearance to the obstacles, because a bad path could be expensive to traverse, and could damage the robot or environment.

While these algorithms were successfully applied in fields such as mobile robots, manipulation planning and human robot planning [1], current virtual environment applications, such as games, pose many new challenges to the algorithms. That is, *natural* paths for many characters traversing in the ever growing environments need to be planned simultaneously and in real-time. Consequently, only a (fraction of a) millisecond per second CPU time may be spent per character for computing the natural path (i.e. a path that is smooth, short, keeps some clearance to obstacles, avoids other characters, *etcetera*).

© Springer-Verlag Berlin Heidelberg 2008

In conclusion, current virtual environments require a fast flexible planner which can generate natural paths. A candidate for the flexible planner is a Potential Field method [3], because it can be used to evade characters and to create smooth paths. Due to the method's local behavior, it will not always find a path.

Roadmap based methods, such as Visibility graphs [1], Probabilistic Roadmap Methods [2], and the A* method operating on a graph-like grid [4], can usually ensure that a path can be found if one exists. However, they lack flexibility because they output a fixed path (extracted from a one-dimensional graph). In addition, the paths are unnatural. While some optimization algorithms exist, they are too slow to be applied in real-time [5].

Recently, the Corridor Map Method (CMM) has been proposed, which satisfies the requirements mentioned above [6]. The CMM directs the global motions of a character traversing a corridor. Such a corridor is extracted from the *corridor map* which is a graph with clearance information. Local motions are controlled by potential fields inside a corridor, providing the desired flexibility.

In this paper, we show how the CMM can be used to simultaneously plan the motions of a large number of characters in real-time. To this end, we first, in Section 2, indicate how to create high-quality corridor maps and how to extract a path efficiently. Next, in Section 3, to obtain more natural motions, we introduce variations in the local behavior of the characters. In particular, we will give a simple approach to get natural lane formation. Also, in Section 4, we describe how the characters can avoid each other. Based on this we introduce in Section 5 two ways of modeling large crowds. The first method uses goal oriented behavior by repeatedly planning paths for the individual characters. The second method uses the corridor map itself to create wandering behavior. Finally, in Section 6, we show how we can plan the motions of coherent groups of characters. Experiments show that we can plan the simultaneous motions of thousands of characters in real-time making the CMM favorable over common A* techniques.

2 The Corridor Map Method

The Corridor Map Method (CMM) consists of an off-line *construction phase* and an on-line *query phase* [6]. These two phases are visualized in Fig. 1.

In the construction phase, a corridor map is created, representing the free space (i.e. the space that is not occupied by the static obstacles) of the environment. The skeleton of the corridors is a graph. We refer the reader to Fig. 1(a) for an example.

A corridor consists of a backbone path and a set of balls centered around this path. A corridor $\mathcal{B} = (B[t], R[t])$ is defined a sequence of maximum clearance balls with radii $R[t]$ whose center points $B[t]$ lie along its backbone path B. The parameter t is an index ranging between 0 and 1, and B is defined as a list of coordinates.

In the query phase, we connect the start and goal position of the character (modeled by a ball with radius r) to the graph and find the shortest backbone path in the graph connecting these positions. From the map, we now extract

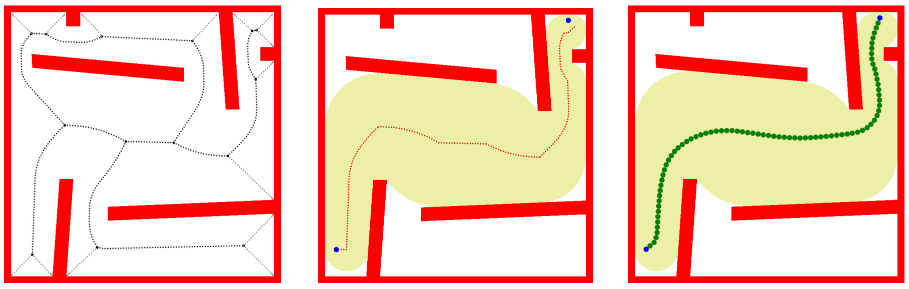

(a) Corridor map (b) Corridor and query (c) Path

Fig. 1. The construction phase (a) and the query phase (b,c) of the CMM

a corridor which is formed by concatenating the corridors corresponding to the edges being part of this path. These steps sre visualized in Fig. 1(b).

While the corridor guides the global motions of the character, its local motions are led by an *attraction point*, $\alpha(x)$, moving on the backbone path of the corridor toward the goal. The attraction point is defined such that making a step toward this point leads the character, located at position x, toward the goal.

The attraction point attracts the character with force $\mathbf{F}_a$. Let d be the Euclidean distance between the character's position and the attraction point $\alpha(x)$. Then $\mathbf{F}_a(x) = f\frac{\alpha(x)-x}{||\alpha(x)-x||}$, where $f = \frac{1}{R[t]-r-d} - \frac{1}{R[t]-r}$. The scalar f is chosen such that f is 0 when the character is positioned on the attraction point. In addition, f is ∞ when the character touches the ball's boundary.

Additional behavior can be incorporated by adding extra forces to $\mathbf{F}_a$, resulting in a force $\mathbf{F}$. The final path is obtained by iteratively integrating $\mathbf{F}$ over time while updating the velocity, position and attraction point of the character. In [6], it is proved that the resulting path is smooth (i.e. C^1-continuous). An example of such a path is displayed in Fig. 1(c).

2.1 Creating High-Quality Corridor Maps

An important impact on the quality of the paths is the quality of the corridor map. In [7], we described an approach to create high-quality maps. The corridors of the map were extracted from the Generalized Voronoi Diagram (GVD) [8].

Using the GVD as basis for the map has three main advantages. First, if a path exists in the free space then it can always be found (because the GVD is a complete representation of the free space). Second, a GVD includes all cycles present in the environment. These cycles provide short global paths and alternative routes which allow for variation in the characters' routes. Third, corridors extracted from the GVD have a maximum clearance. Such a corridor provides maximum local flexibility.

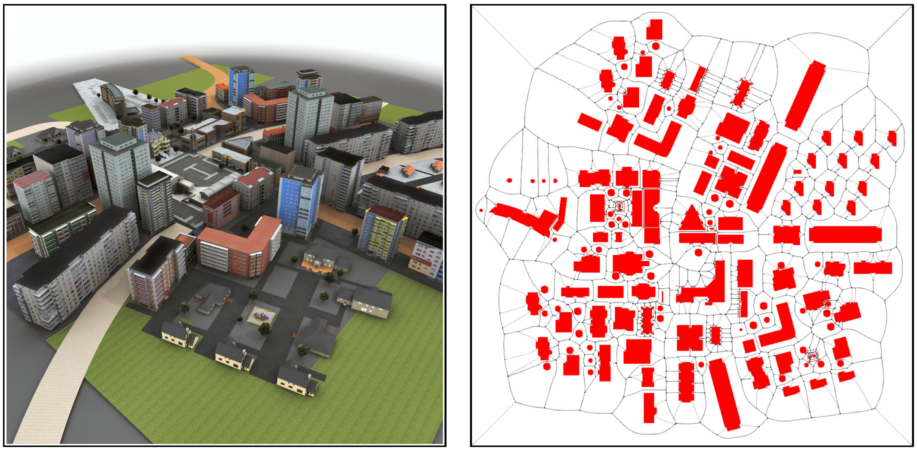

(a) Environment (b) Footprint and corridor map

Fig. 2. The City environment. While its geometry is three-dimensional, its footprint (used for generating the corridor map) is only two-dimensional.

To keep the map small and to improve the running times in the query phase, we sampled the corridors such that the coverage of the free space was still large while the efficiency in the query phase was high.

As an example, consider Fig. 2 which shows our test environment. The city measures 500x500 meter. Its geometry, displayed in Fig. 2.1, is composed of 220K triangles. Using this number of triangles will lead to a low performance. Instead, we used its footprint (2,122 triangles) to generate the corridor map, see Fig. 2.1. This took 0.64 seconds on a PC with a NVIDIA GeForce 8800 GTX graphics card and an Intel Core2 Quad CPU (2.4 GHz) with 4 GB memory. (While this processor has four CPU's, our application, implemented in Visual C++ and run under Windows XP, used only one core.) The map comprised 1,434 vertices, 1,606 edges and 25,863 samples (i.e. the total number of balls in all corridors).

2.2 Fast Extraction of a Corridor

In [7], we elaborated on how to facilitate efficient extraction of corridors and paths. We showed that the average extraction time for a corridor (excluding the times for finding the shortest backbone path and connecting the query) was 0.87 ms in the City environment. (We took the average of 10,000 random corridors.)

We looked at two different algorithms for finding the shortest corridor enclosing the query. Experiments showed that Dijkstra's algorithm increased the time by 138.2% while A* increased the time by only 11.5%. Then, we looked at two different approaches for connecting a query to the corridor map. Using linear search increased the time by 59.7% while using a search structure based on a *kd*-tree only increased the time by 1.0%. In conclusion, by using A* and a *kd*-tree, the average extraction time of a corridor was low, i.e. 1.19 ms.

2.3 Fast Extraction of a Path

We wanted to know how fast we could compute a path. We extracted 10,000 paths in the City environment and recorded the average running time. Experimental results showed that computing a path took 1.8 ms (including the 1.19 ms for computing a corridor). To view this result in the right perspective, we defined the CPU-load. That is, the CPU-load is the total CPU time / averaged traversed time $*$ 100%. Since the averaged traversed time of a character (walking at 1.2 m/s) was 260 seconds, the CPU-load is 0.00069%. Hence, the CMM can be used for steering many characters simultaneously. This will be discussed further in Section 5 and 6.

3 Path Variation

The original CMM can be easily extended to create high-quality alternative paths that a character can follow within a corridor. This not only presents a more challenging and less predictable opponent for the player, but also enhances the realism of the gaming experience.

To generate slightly different paths every time the same path planning problem has to be solved, a random force (bias) is added to the attractive force $\mathbf{F}_a$. A coherent-noise function like Perlin Noise [9] can be used to control the direction of the bias, ensuring that it will change smoothly at every step of the integration.

In our problem setting, Perlin Noise is implemented as a 2D function. The current attraction point $\alpha(x)$ is given as an input and a direction for the bias is computed which varies pseudo-randomly. Let θ denotes this direction expressed as an angle of rotation with respect to the current attractive vector, i.e. $\alpha(x)-x$. Then, the random force can be defined as:

$$\mathbf{F}_{rand} = c_{rand}\ \mathcal{R}(\theta)\frac{\alpha(x)-x}{||\alpha(x)-x||},$$

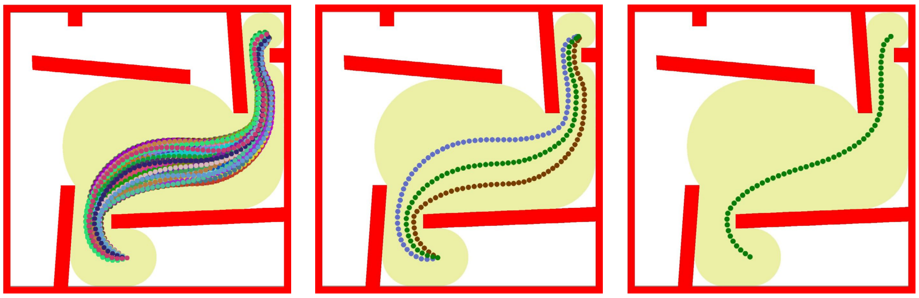
where c_{rand} is a small constant specifying the relative strength of the force, $\mathcal{R}(\theta)$ represents the 2D rotation matrix and the angle $\theta \in [-\pi/2, +\pi/2]$.

The quality of the computed path is affected by the frequency of the noise function. A too high frequency noise leads to the retrieval of unrealistic paths, since the character will change its direction at almost every successive time step. On the contrary, a low frequency results in smooth changes, generating aesthetically pleasant paths like the ones depicted in Fig. 3.

As individuals usually show a preference for certain paths over others, deterministic variations of paths are also necessary to simulate behaviors that have been widely noted in the crowd and pedestrian literature.

For example, lanes can be formed on either side of the corridor by exerting at every time step a force perpendicular to the attractive force, hence $\theta \in \{-\pi/2, +\pi/2\}$. A positive angle steers the character to the left of the backbone path, whereas a negative one to the right. The perpendicular force is defined as:

$$\mathbf{F}_{perp} = c_{perp}\ \mathcal{R}(\theta)\frac{\alpha(x)-x}{||\alpha(x)-x||},$$



(a) 100 random paths (b) Lane formation inside the corridor (c) Following a shorter and more direct path

Fig. 3. Generating high-quality alternative paths using the CMM

where $c_{perp} = k\frac{R[t]-2r}{d}$ and k is a constant used to specify how close to the boundary of the corridor the character moves.

We refer the reader to Fig. 3 for a graphical representation of the described technique. Different left/right paths are computed by varying the value of the constant k.

Another example, resulting in a more natural and visually interesting movement, is to generate paths that either take tight or wide corners. In this approach the direction of the backbone path is used to bias the character's motion [10]. The character looks intelligent, in the sense that it anticipates the direction of the path it has to follow and starts to turn in advance. A convincing path is computed as can be examined in Fig. 3.

3.1 Results

We implemented the presented approaches to test their applicability in real-time applications like computer games. Since the proposed methods can be computed efficiently, which was experimentally confirmed in [10], they influence the running time of the algorithm marginally. Consequently, alternative paths can be computed in real-time.

Apart from the performance, we are also interested in the quality of the resulting paths. Experiments in [10] have indicated that our proposed techniques can successfully generate alternative routes that are aesthetically pleasant to the observer.

The first method uses a noise function to generate slightly different paths in response to a given query. As expected, the computed paths have almost the same clearance and length as the smooth path computed by the original CMM.

The second method forms different lanes on either side of the extracted corridor. Although the computed paths can get close to the boundary of the corridor, they still keep a safe amount of clearance. In addition, no significant difference in the length of the paths is observed.

In the third method the character takes shortcuts through the turns of the backbone path, trying to avoid any unnecessary detour and minimize the time needed to reach its goal. Thus, a shorter and more direct path is generated.

In conclusion, our techniques extend the basic functionality of the CMM by creating in real-time high-quality alternative paths. Combined with existing agent-based models the proposed methods lead to convincing characters that exhibit human-like path planning as discussed below.

4 Obstacle Avoidance

When we are dealing with more than one character, we have to choose an appropriate force such that characters evade each other naturally.

We use a part of Helbing and Molnár's social force model for collisions avoidance because their simulations have shown that it exhibits realistic crowd behavior [11]. The model describes three force functions which represent the acceleration toward the desired velocity of motion, the behavior of characters keeping a certain distance to other characters and borders, and attractive effects among characters. The force $\mathbf{F}_a$ described in Section 2 captures the effect of their acceleration force and the force keeping characters away from borders. As collision avoidance force, $\mathbf{F}_c$, we use the force described in equation (3) of their paper. Unlike [11], we do not model attractive effects among characters.

In the computation of force $\mathbf{F}_c$, we have to find the set of neighbors for each character. Since this operation is carried out many times, we have to revert to an efficient data structure for answering nearest neighbors queries. We maintain a 2D grid storing the locations of all characters (with radius r). Each cell in the grid stores a sorted set of character id's. Preliminary experiments have shown that the cell size c can be chosen fairly small, e.g. $c = 3 + 2r$ meter, to obtain realistic collision avoidance. By including the term $2r$, we only have to check 3 by 3 cells for carrying out a nearest neighbors query. Also, an update corresponding to a changed position of a character is efficient since we use a set.

5 Crowd Simulation

Real-time crowd simulation has gained much attention recently [11, 12, 13, 14]. It requires the modeling of group behavior, pedestrian dynamics, path planning and graphical rendering [13]. We focus on the path planning part.

We model two types of behavior for the characters, i.e. *goal-oriented* and *wandering* behavior. In Section 5.1, we discuss how to model a crowd in which each character has its own (long term) goal. Next, we elaborate on wandering behavior in Section 5.2 which comprises making decisions more locally.

5.1 Goal Oriented Behavior

Goal oriented behavior can be achieved easily by assigning each character a random start and goal position. These positions will fix a corridor. When a character has reached its goal, a new goal will be chosen, *ad infinitum*.

If we only used force $\mathbf{F}_a$ to guide characters toward their goals (and force $\mathbf{F}_c$ to avoid each other), they would have the tendency to clutter up around

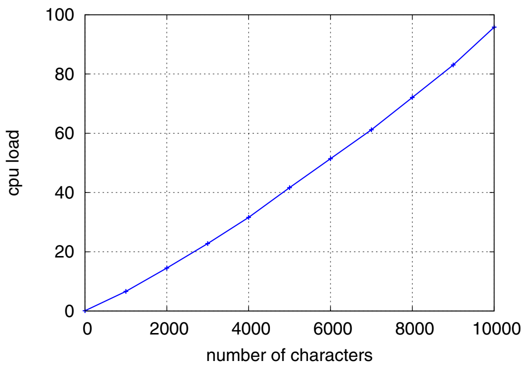


Fig. 4. The relation between the number of characters in the crowd simulation and the CPU-load

their backbone paths. To enforce a nice spread (and path variation) inside the corridors, we add the force $\mathbf{F}_{perp}$ biasing a character to the right with respect to its desired direction.

The simulation consists of some user defined number of iterations. For each character, the task per iteration includes computing the forces, integrating the final force $\mathbf{F} = \mathbf{F}_a + \mathbf{F}_c + \mathbf{F}_{perp}$, adding the new position of the character to its path, and possibly choosing a new goal (and corresponding corridor).

Results. We integrated force $\mathbf{F}$ with Verlet integration with step size $\Delta t = 0.1$ and set the maximum speed to 1.2 m/s [15]. The cell size of the nearest neighbor grid was set to $c = 3 + 2r = 3.5$ meter. In the experiments, we measured the CPU-load for a varying number of characters. When the CPU-load was at most 100%, we considered the performance as real-time.

We simulated crowds in the City Environment from Fig. 2 with a varying number of characters (with a radius of 25 cm, i.e. $r = 0.25$). Fig. 4 shows the performance of our application for crowds with up to 10,000 characters. The figure makes clear that approximately 10,000 characters can be simulated in real-time. In addition, the performance does not degrade significantly when the environment becomes rather crowded. We conclude that the CMM can be used for real-time path planning with many characters.

5.2 Wandering Behavior

Rather than using the corridor map to plan paths for each of the characters in the crowd, there is also an alternative approach in which the characters simply wander around. The idea is as follows. As before, for a character at position x there is an attraction point $\alpha(x)$. This attraction point lies on some edge ϵ of the graph and we assume that we have decided on a direction along which the attraction point will move along ϵ toward a vertex ν. We compute forces as before and move the character accordingly. Next, we update the attraction point by moving it further along ϵ toward ν. If the attraction point reaches ν, we randomly pick one of the outgoing edges ϵ' of vertex ν, unequal to edge ϵ, and continue moving the attraction point along edge ϵ'. So the attraction point will follow some random walk through the graph and the character follows the attraction point.

There is one complication. When ν is a dead end in the graph, that is, it has only edge ϵ as an outgoing edge, the attraction point must move backwards along the same edge. However, because of the way attraction points are defined (i.e. the furthest point for which x still lies in the clearance disk) the result is that the attraction point jumps far back along the edge, and, hence, the characters will not even get close to vertex ν, leaving part of the dead end in the environment void of characters. To remedy this we make a distinction between short dead ends and long dead ends. We recursively remove short edges that represent dead ends. For long dead ends we create some invisible obstacle at the end vertex ν and create a corridor around it. As a result, characters will move till the end of the dead end, move around the invisible obstacle, and return again along the edge.

The method is very fast because no searches in the graph are required. Preliminary experiments show, in particular when combined with lane formation as described in Section 3, that the approach leads to natural wandering behavior. We can extend the method by giving certain edges preference over other edges and choosing the random continuation edge ϵ' taking these preferences into account. The method can also be combined easily with other characters that do have goal oriented behavior.

6 Coherent Groups

A virtual environment, such as a city, is usually populated with many characters which often operate in groups. Take for example a guided tour through the city. Here, each group member needs to stay in close proximity to other members while moving from one location in the city to another. The CMM enables us to plan such paths very efficiently [16].

For a coherent group of characters, the distances between the characters need to be limited. This can be achieved in two directions, i.e. the *lateral* and *longitudinal direction*. The lateral direction is the direction perpendicular to the direction of movement, i.e. the direction of the backbone path. The longitudinal direction is the direction along the backbone path. We refer to the separation of the characters in the lateral and longitudinal directions as the lateral dispersion and the longitudinal dispersion, respectively. Please note, if the dispersion increases, the coherence decreases.

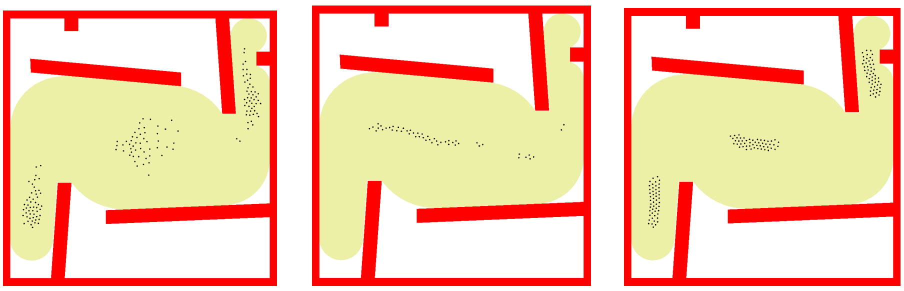
By using the CMM to create a corridor for the group, the lateral coherence is bounded, namely by the radius $R[t]$ of the clearance balls on every point on the backbone path. Since no character can leave the corridor, the maximum lateral distance between any two characters is limited by the maximum radius of all clearance balls along the backbone path. However, this maximum radius might be too large. Therefore, we introduce a constant g_w that represents the maximum width of the coherence group. Using this constant we adapt the corridor $\mathcal{B}$ to $\mathcal{B}^* = (B[t], \max\{R[t], g_w\})$.

The longitudinal dispersion can be bounded in a different manner. Let us introduce another constant, g_A, called the group area, which represents the area in the corridor that the group is allowed to occupy. The group area is defined as the union of clearance balls from a start point to an end point on the backbone path. The start point is defined as the center of the furthest advanced ball in which the least advanced character is enclosed. The end point is defined as the center of the least advanced ball in which the furthest advanced character is enclosed. Now, we define the minimum attraction point, α_{min}, as the least advanced attraction point of all attraction points, $\alpha(x_i)$, of the characters in the group. By definition, this is the start point of the group area. The maximum allowed attraction point, α_{max}, is defined as the point on the backbone path such that the union of balls from α_{min} to α_{max} is exactly g_A. If an attraction point $\alpha(x)$ for any character is further advanced on the backbone path than the maximum attraction point α_{max}, the maximum attraction point is used as the attraction point of the character. Consequently, characters that are in front of the group will be attracted backward, making them wait for the rest.

By varying the two parameters, g_w and g_A, we can influence the behavior of the group. That is, high values result in a weak coherent group while small values result in a strong coherent group. Please note that the area should not be chosen too small, otherwise the characters will not fit inside this area.

6.1 Results

We generated several paths for a group of 50 characters with varying degrees of coherence. The first series of paths were created with a large value for both g_w and g_A ($g_w = 30$, $g_A = 1000$). A snapshot of the paths is depicted in Fig. 5(a).



(a) Three snapshots of the group at different stages on the paths. The width of the corridor is $g_w = 30$, the group area is $g_A = 1000$

(b) A single snapshot of the group. The width of the corridor is $g_w = 3$, the group area is $g_A = 1000$

(c) Three snapshots of the group at different stages on the paths. The width of the corridor is $g_w = 3$, the group area is $g_A = 60$

Fig. 5. The effect on different lateral and longitudinal dispersions on 50 characters

This results in a weak coherent group, where the characters are scattered over the whole corridor. By decreasing the width of the corridor ($g_w = 3$, $g_A = 1000$) the group becomes more coherent in the lateral direction, see Fig. 5(b). However, the longitudinal coherence is weak. By also decreasing the area of the group ($g_w = 3$, $g_A = 60$) the group becomes more coherent in both directions, see Fig. 5(c).

7 Conclusions and Future Work

In this paper we have described how the recently introduced Corridor Map Method (CMM) can be used to plan the motions of thousands of characters in a virtual world in real-time. We considered crowds of wandering characters without clear goals, characters with individual goals, and groups of characters.

The advantage of using the CMM is that it is fast, it is flexible, and it produces natural paths. Because of the flexibility it is easy to introduce additional constraints on the paths of the characters. For example, we can incorporate all three types of motions simultaneously. Also, we can add additional static or dynamic obstacles that must be avoided. In addition, we can incorporate personal preferences on the motions of the characters, for example to improve the animations.

Currently, we are investigating how we can incorporate influence regions in the environment. Such regions can either be dangerous places the characters prefer to avoid or nice places that they prefer to visit. Also we work on techniques to improve the way the characters avoid each other by extending the Helbing model. And finally, we are experimentally validating the model and approach by observing real people using cameras and motion capture equipment. We expect this to lead to even more natural character motions.

Acknowledgements

This research has been supported by the GATE project, funded by the Netherlands Organization for Scientific Research (NWO) and the Netherlands ICT Research and Innovation Authority (ICT Regie). In addition, part of this research has been funded by the Dutch BSIK/BRICKS Project.

References

1. Latombe, J.-C.: Robot Motion Planning. Kluwer, Dordrecht (1991)
2. LaValle, S.: Planning Algorithms (2006), `http://planning.cs.uiuc.edu`
3. Rimon, E., Koditschek, D.: Exact robot navigation using artificial potential fields. IEEE Transactions on Robotics and Automation 8, 501–518 (1992)
4. Hart, P., Nilsson, N., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics 4, 100–107 (1968)

5. Geraerts, R., Overmars, M.: Creating high-quality paths for motion planning. International Journal of Robotics Research 26, 845–863 (2007)
6. Geraerts, R., Overmars, M.: The corridor map method: A general framework for real-time high-quality path planning. Computer Animation and Virtual Worlds 18, 107–119 (2007)
7. Geraerts, R., Overmars, M.: Enhancing corridor maps for real-time path planning in virtual environments. In: Computer Animation and Social Agents (2008)
8. Hoff, K., Culver, T., Keyser, J., Lin, M., Manocha, D.: Interactive motion planning using hardware-accelerated computation of generalized Voronoi diagrams. In: IEEE International Conference on Robotics and Automation, pp. 2931–2937 (2000)
9. Perlin, K.: An image synthesizer. Computer Graphics 19(3), 287–296 (1985); SIGGRAPH 1985 Proceedings
10. Karamouzas, I., Overmars, M.H.: Adding variation to path planning. In: Computer Animation and Social Agents (2008)
11. Helbing, D., Molnár, P.: Social force model for pedestrian dynamics. Physical Review 51, 4282–4287 (1995)
12. Morini, F., Yersina, B., Maïm, J., Thalmann, D.: Real-time scalable motion planning for crowds. In: International Conference on Cyberworlds, pp. 144–151 (2007)
13. Sud, A., Gayle, R., Andersen, E., Guy, S., Lin, M., Manocha, D.: Real-time navigation of independent agents using adaptive roadmaps. In: ACM symposium on Virtual reality software and technology, pp. 99–106 (2007)
14. Treuille, A., Cooper, S., Popović, Z.: Continuum crowds. Transactions on Graphics 25, 1160–1168 (2006)
15. Knoblauch, R., Pietrucha, M., Nitzburg, M.: Field studies of pedestrian walking speed and start-up time. Transportation Research Record, 27–38 (1996)
16. Kamphuis, A., Overmars, M.: Finding paths for coherent groups using clearance. In: Eurographics/ ACM SIGGRAPH Symposium on Computer Animation, pp. 19–28 (2004)

Real-Time Path Planning and Navigation for Multi-agent and Crowd Simulations

Ming C. Lin, Avneesh Sud[⋆], Jur Van den Berg, Russell Gayle, Sean Curtis, Hengchin Yeh, Stephen Guy, Eric Andersen[⋆⋆], Sachin Patil, Jason Sewall, and Dinesh Manocha

Department of Computer Science
University of North Carolina
Chapel Hill, NC, 27599-3175 U.S.A.
{lin,sud,berg,rgayle,seanc,yero,sjguy, andersen,sachin,sewall,dm}@cs.unc.edu
http://gamma.cs.unc.edu/

Abstract. We survey some of our recent work on real-time path planning and navigation of multiple autonomous agents in dynamic environments. The driving application of our work are real-time crowd simulation for computer games, virtual environments, and avatar-based online 3D social networks. We also present extensions to these methods for accelerating the overall simulation and for modeling more complex behaviors. Finally, we present some preliminary results from our simulations.

Keywords: Velocity Obstacles, Agent-based Simulation, Roadmaps.

1 Introduction

Modeling of crowds, multiple agents and swarm-like behaviors has been widely studied in computer graphics, robotics, architecture, physics, psychology, social sciences, and civil and traffic engineering. Swarms and crowds, ubiquitous in the real world from groups of humans to schools of fish, are vital phenomena to understand and model. In this paper, we address the problems of collision-free path computation for multiple independent agents moving in a dynamic complex game environment and a closely related problem of local navigation among multiple moving agents.

Agent-based techniques focus on modeling individual behaviors and intents. They offer many attractive benefits, as they often result in more realistic and detailed simulations. However, individuals constantly adjust their behavior according to dynamic factors (e.g. another approaching individual) in the environment. Therefore, one of the key challenges in a large-scale agent-based simulation is global collision-free path planning for each virtual agent. The path planning

[⋆] Now at Microsoft Coproration.

[⋆⋆] Now at University of Washington.

© Springer-Verlag Berlin Heidelberg 2008

problem can become very challenging for real-time applications with a large group of moving agents, as each is a dynamic obstacle for others. Moreover, there may be other dynamic obstacles in the scene, and the underlying applications cannot make any assumptions about their motion. Many prior techniques are either restricted to static environments, or only perform local collision avoidance computations, or may not scale to complex environments with hundreds of agents. The use of solely local methods can result in unnatural behavior or "getting stuck" in local minima. As a result, it is important to combine them with global methods in order to guarantee path convergence for each agent. These problems tend to be more challenging in a dynamically changing scene with multiple moving virtual agents.

We present several complementary approaches for path planning and navigation of multiple virtual agents in a dynamic environment. These methods can be applied to interactive crowd simulation and coordination of multiple autonomous agents in computer games. These include (a) Multi-agent Navigation Graph (MaNG) [37]; (b) Adaptive Elastic ROadmaps (AERO) [38]; and (c) navigation using Reciprocal Velocity Obstacles (RVO) [44]. Each of these methods are suitable for various environments of different characteristics. We also demonstrate how these algorithms can be accelerated by "Pedestrian Levels of Detail" (PLODs) and extended to capture complex human behaviors using "Composite Agents" (CA).

2 Related Work

In this section, we give a brief overview of prior work in this area.

Multiple Moving Entities: The problem of motion planning among multiple agents moving simultaneously in an environment is challenging because of the addition of the number of degrees of freedom which becomes large. The problem was proved to be intractable [18]. It is a specific case of the general time-varying motion planning problem. Two key approaches are known to address this problem: centralized and decoupled planning.

The *centralized* approaches [17,18] consider the sum of all the robots as a single one. For that, configuration spaces of individual robots are combined (using Cartesian product) in a composite one, in which a solution is searched. As the dimension of the composite space grows with the number of degrees of freedom added by each entity, the problem complexity becomes prohibitively high.

Contrarily, the *decoupled* planners proceed in a distributed manner and coordination is often handled by exploring a *coordination space*, which represents the parameters along each specific robot path. Decoupled approaches [34,45] are much faster than centralized methods, but may not be able to guarantee completeness.

Dynamic Environments: Evolving elements significantly increase the difficulty of the motion planning problem. In fact, motion planning for a single disc with bounded velocity among rotating obstacles is PSPACE-hard [18].

However, there have been many attempts to provide practical methods to cope with changing environments. For example, Stentz et al. proposed the D* deterministic planning algorithm to repair previous solutions instead of re-planning from scratch [15,35].

There are two main approaches for adapting randomized planners to dynamic environments [10,19]. The first one includes both PRMs and RRTs that reuse previously computed information to aid in finding a new path [5,11,12,20]. The second integrates obstacle motion directly into the planning process. Some variations plan directly in a C-space augmented with a time parameter [28].

Rather than changing the roadmap, other work on dynamic environments has focused on adjusting or modifying the path. Potential field planners use gradient descent to move toward a goal, at a potential sink [14]. Building on these ideas, several variation of dynamic changing or elastic roadmaps have been proposed [8,29,46].

Agent Simulation and Crowd Dynamics: Modeling of collective behaviors has been heavily studied in many fields [9,13,23,32,36]. Simulation of multiple avatars agents and crowds have also been well studied in graphics and VR [1,30,33,41,42]. They differ based on problem decomposition (discrete vs continuous), stochastic vs deterministic, etc.

Numerous approaches have been proposed using variants of agent-based methods, rule-based techniques, and social force models [4,22,24,25,26,31,39,43]. They focus mainly on local planning of agents, and can exhibit emergent behaviors. Global methods using path planning algorithms have also been suggested [2,6,16,27,33,40], but are mostly limited to static environments. More recently algorithms have been proposed to extend the roadmap-based methods to dynamic environments and multiple agents [7,8,21,27,47] in relatively simple environments with only a few entities. Hybrid methods that treat each agent as a particle in a dynamical system combined with high-level planning can lead to instability in the simulation or entrapment at local minima in the energy potential (See examples discussed in [42]).

3 Multi-agent Navigation Graphs

We introduce a new data structure called "multi-agent navigation graph" or MaNG and compute it efficiently using GPU-accelerated discrete Voronoi diagrams. Voronoi diagrams have been widely used for path planning computations in static environments and we extend these approaches to dynamic environments. Voronoi diagrams encode the connectivity of the free space and provide a path of maximal clearance for an agent from other obstacles. In order to use them for multiple moving agents in a dynamic scene, prior approaches compute the Voronoi diagram for each agent separately, treating the other agents and the environment as obstacles. This approach can become very costly for large number of virtual agents. Instead, we compute the *second order* Voronoi diagram of all the obstacles and agents, and show that the second order Voronoi diagram provides *pairwise* proximity information for all the agents simultaneously.

Therefore, we combine the first and second order Voronoi graphs to compute a single MaNG which provides global path planning of multiple virtual agents.

We describe novel algorithm for computing the first and second order diagrams of agents and obstacles using graphics hardware. In practice, our approach can handle environments with few hundred agents and obstacles in real time. We make no assumption on the motion, and the computed path lies along the Voronoi boundary, i.e. farthest from the obstacles. We also observe many interesting and emergent behaviors based on our method [37].

4 Heterogeneous Crowd Simulations Using AERO

We present a new algorithm for real-time simulation of large-scale heterogeneous crowds in complex dynamic environments. In his pioneering work, Gustave Le Bon [3] defined a *heterogeneous crowd* as consisting of many dissimilar types of groups, each with potentially independent behavior characteristics and goals. Examples include large exposition halls, wide festival arenas, busy urban street scenes, etc. Real-time simulation of heterogeneous crowds is highly challenging because pedestrian dynamics exhibits a rich variety of both independent and collective effects, such as lane formations, oscillations at bottlenecks, chemotaxis and panic effects.

Our approach is based on a novel concept called *"Adaptive Elastic ROadmaps" (AERO)* [38]. It is a connectivity graph structure that is lazily computed using a generalized crowd dynamics model. Specifically, we use a reactive roadmap computation algorithm [8] that updates the links of the roadmap graphs using a particle-based dynamics simulator. This algorithm includes path modification, addition and deletion of links, as well as multiple lane formations around each link. Our dynamics model simultaneously captures macroscopic mutual interaction among multiple moving *groups* and microscopic local forces among *individual* pedestrians or agents. We use AERO to perform dynamic, global path planning based on this force model. This approach has been successfully demonstrated in complex urban scenes, trade-shows as well as terrain scenarios with multiple moving vehicles.

5 Navigation Using Reciprocal Velocity Obstacles

We also present another technique for local navigation. Our approach is decomposed into two levels. The higher-level deals with the global path planning towards the goal using a precomputed roadmap [18], and the lower-level addresses local collision avoidance and navigation using Reciprocal Velocity Obstacles (RVO) [44]. We assume that the other agents are dynamic obstacles whose future motions are predicted as linear extrapolations of their current velocities. RVO provides a principle to select a velocity for agent A_i and implicitly assumes that the other agents A_j use similar collision avoidance reasoning. Informally speaking, this means that agent A_i does only half of the effort to avoid a collision with agent A_j, and assumes that the other agent will take care of the other half.

The Reciprocal Velocity Obstacle $RVO_j^i(\mathbf{v}_j, \mathbf{v}_i)$ of agent A_j to agent A_i is defined as the set consisting of all those velocities $\mathbf{v}_i$ for A_i that will result in a collision at some moment in time with agent A_j, *if* agent A_j chooses a velocity in its Reciprocal Velocity Obstacle as well. n other words, if A_i chooses its velocity outside the Reciprocal Velocity Obstacle, A_i is guaranteed to avoid collisions with agent A_j, *provided* that A_j applies the same navigation algorithm, and chooses a new velocity outside its Reciprocal Velocity Obstacle. As there are multiple agents around, each agent computes its Reciprocal Velocity Obstacle as the *union* of the Reciprocal Velocity Obstacles corresponding to the individual agents, and selects a velocity outside this union to avoid collisions. Also, the Reciprocal Velocity Obstacle method is guaranteed to avoid oscillatory behavior of the agents. This basic algorithmic framework can also incorporate kinodynamic constraints, orientation, and visibility regions of each agents, etc.

6 Extensions

In this section, we present techniques to extend these methods for capturing more complex behaviors and accelerate the overall simulations.

6.1 Modeling Complex Behaviors

We introduce the notion of composite agents to effectively model different avatar behaviors for agent-based crowd simulation. Each composite agent consists of a basic agent that is associated with one or more *proxy agents.* A proxy agent possesses the same kind of external properties as an agent, i.e. if every agent's external property consists of the velocity, then the proxy agent must have its own velocity as well. The values of these external properties, can be different, e.g. the proxy can possesses a velocity that is different from anyone else. The external properties of a proxy P_j is denoted as ε_j. The internal state ι_j of a proxy agent, however, need not be the same set of properties of the basic agent. We also define that P_j has access to the internal state ι_i of its parent A_i. We denote the set of all proxy agents being simulated as $Proxies = \bigcup_i proxy(A_i)$

This formulation allows an agent with given physical properties to exercise its influence over other agents and the environment. Composite agents can be added to most agent-based simulation systems and used to model emergent behaviors among individuals. These behaviors include aggression, impatience, intimidation, leadership, trailblazing and approach-avoidance conflict, etc. in complex scenes. In practice, there is negligible overhead of introducing composite agents in the simulation.

6.2 Accelerating Simulations

We can accelerate crowd simulation by using "Pedestrian Levels of Detail" (PLODs). PLODs is a new hierarchical data structure based on Kd-tree

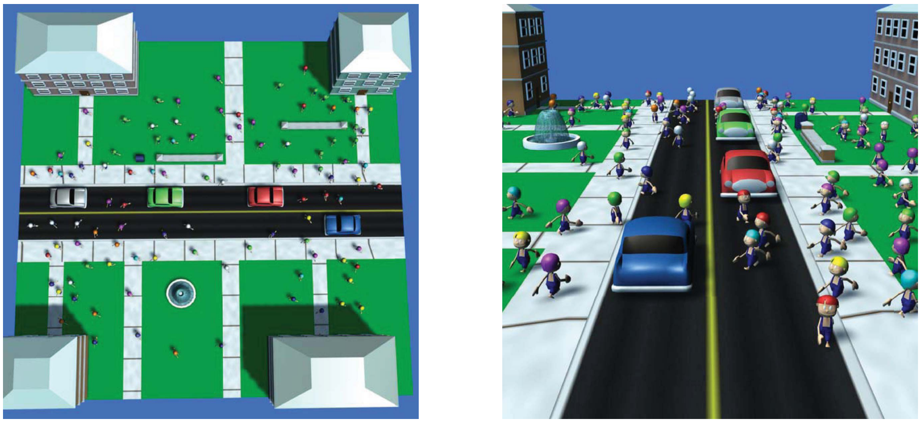


Fig. 1. Multi-agent simulation using MaNG

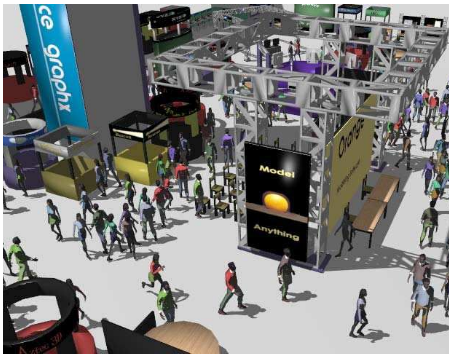


Fig. 2. Real-time heterogeneous crowd simulation using AERO

selectively and dynamically computed. It is used to adaptively cluster agents to accelerate large-scale simulation of heterogeneous crowds. Our formulation is based on the observation and empirical validation in traffic engineering that crowds exhibit self-organization in pedestrian flows. Depending on their dynamic states (e.g. walking speed, heading directions), spatial proximity, and behavior characteristics, agents in heterogeneous crowds are adaptively grouped and subdivided into PLODs, thus reducing the overall computation of heterogeneous crowd simulation.

7 Results

We demonstrate our approaches on several complex indoor and outdoor scenarios. In the first scenario, we show that we can compute MaNG for real-time path planning of 100 – 200 agents in real time (approx. 10 fps) in a small town environment (see Fig. 1). In the second scenario shown in Fig. 2, we illustrate the

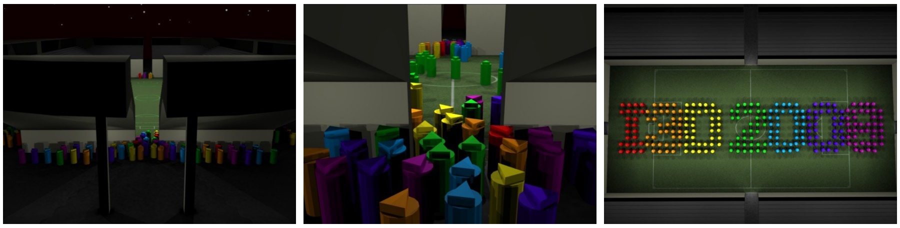


Fig. 3. Real-time Navigation using RVO. 250 agents form the word 'I3D 2008' on the field after entering the stadium; congestion develops near the stadium entrances.

performance of our real-time heterogeneous crowd simulation using AERO on a exhibition Hall scene, consisting of $1,000$ pedestrians and 500 booths. Fig. 3 shows pedestrians crossing a street at four corners of crosswalks, demonstrating "lane formation" that naturally emerges using RVO. Fig. 4 demonstrates a variety of behaviors generated using our novel framework of *Composite Agents* on top of any agent-based simulation.

Each one of these methods have its advantages and limitations. The choice of methods depends to some degree on the complexity of the simulation environments, the density of agents and their physical interactions, the diversity of their behaviors, etc.

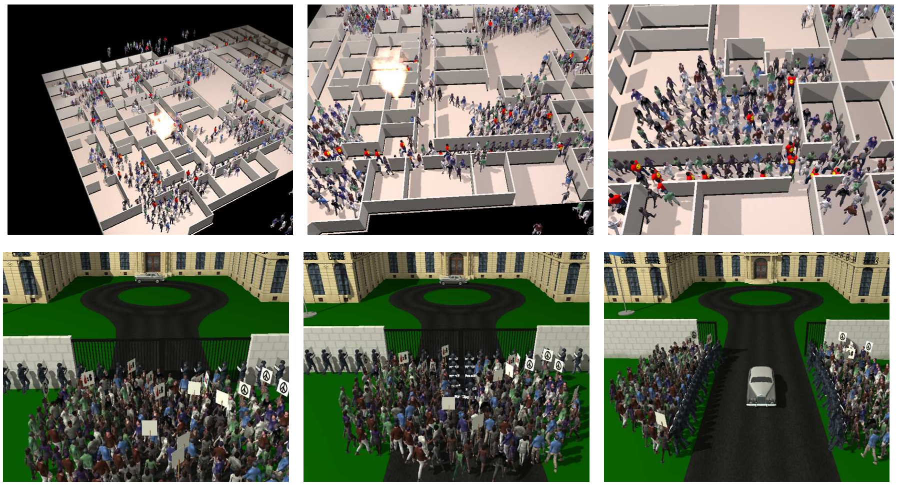


Fig. 4. Modeling of Complex Behaviors. Top row depicts an emergency evacuation in an office building. The agents in red are aggressive agents. They are able to carve their own way through the crowd and exit more quickly than the others. The bottom row simulates a crowd of protesters outside an embassy. Two files of policemen clear the protesters off the road. Notice that when forcing their way into the crowd, even if the actual gap between the individual policemen is enough for a protest or to pass through, the perceived continuity of authority prevents the protesters from breaking the barricade.

8 Discussion

Among these methods, MaNG is better suited for a smaller scene with hundreds of autonomous agents running on a desktop PC with a top-of-the-line GPU. MaNG works well for all types of environments, including open outdoor terrains.

AERO runs at interactive rates for an environment with upto thousands of agents. Some performance gain can be achieved by further exploiting GPUs to compute the interaction forces among agents. AERO currently uses a physics-based social force model. However, a more geometric, velocity-based local navigation method, such as RVO, can be used instead. AERO performs very well for structured urban scenes and modestly well for open landscapes; while the accompanying acceleration method, PLOD, achieves most gain on structured landscapes.

Our current planner that uses RVO as a low-level collision avoidance module is built upon a static, precomputed roadmap. This planning algorithm works best for structured environments. The algorithm scales well as the number of cores increases, thus highly parallelizable to run on many-core architectures, similarly for composite agents.

Acknowledgments. The research presented here is supported in part by Army Research Office, Intel, National Science Foundation, and DARPA/RDECOM.

References

1. Ashida, K., Lee, S.J., Allbeck, J., Sun, H., Badler, N., Metaxas, D.: Pedestrians: Creating agent behaviors through statistical analysis of observation data. In: Proc. Computer Animation (2001)
2. Bayazit, O.B., Lien, J.-M., Amato, N.M.: Better group behaviors in complex environments with global roadmaps. In: Int. Conf. on the Sim. and Syn. of Living Sys. (Alife), pp. 362–370 (2002)
3. Le Bon, G.: The Crowd: A Study of the Popular Mind. Dover Publications (1895)
4. Cordeiro, O.C., Braun, A., Silveria, C.B., Musse, S.R., Cavalheiro, G.G.: Concurrency on social forces simulation model. In: First International Workshop on Crowd Simulation (2005)
5. Ferguson, D., Kalra, N., Stentz, A.: Replanning with RRTs. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (May 2006)
6. Funge, J., Tu, X., Terzopoulos, D.: Cognitive modeling: Knowledge, reasoning and planning for intelligent characters. In: Proc. of ACM SIGGRAPH, pp. 29–38 (1999)
7. Garaerts, R., Overmars, M.H.: The corridor map method: Real-time high-quality path planning. In: ICRA, Roma, Italy, pp. 1023–1028 (2007)
8. Gayle, R., Sud, A., Lin, M., Manocha, D.: Reactive deformation roadmaps: Motion planning of multiple robots in dynamic environments. In: Proc IEEE International Conference on Intelligent Robots and Systems (2007)
9. Helbing, D., Buzna, L., Werner, T.: Self-organized pedestrian crowd dynamics and design solutions. Traffic Forum 12 (2003)
10. Hsu, D., Kindel, R., Latombe, J.-C., Rock, S.: Randomized kinodynamic motion planning with moving obstacles. International Journal of Robotics Research (2002)

11. Jaillet, L., Simeon, T.: A PRM-based motion planning for dynamically changing environments. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2004)
12. Kallmann, M., Mataric, M.: Motion planning using dynamic roadmaps. In: Proceedings of the IEEE Conference on Robotics and Automation (ICRA) (April 2004)
13. Kamphuis, A., Overmars, M.: Finding paths for coherent groups using clearance. In: Proc. of ACM SIGGRAPH / Eurographics Symposium on Computer Animation, pp. 19–28 (2004)
14. Khatib, O.: Real-time obstable avoidance for manipulators and mobile robots. IJRR 5(1), 90–98 (1986)
15. Koenig, S., Likhachev, M.: Improved fast replanning for robot navigation in unknown terrain. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (May 2002)
16. Lamarche, F., Donikian, S.: Crowd of virtual humans: a new approach for real time navigation in complex and structured environments. Computer Graphics Forum 23(3), 509–518 (2004)
17. Latombe, J.C.: Robot Motion Planning. Kluwer Academic Publishers, Dordrecht (1991)
18. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006), http://msl.cs.uiuc.edu/planning/
19. La Valle, S., Kuffner, J.: Randomized kinodynamic planning. International Journal of Robotics Research (2001)
20. Leven, P., Hutchinson, S.: Toward real-time path planning in changing environments. In: Proceedings of the fourth International Workshop on the Algorithmic Foundations of Robotics (WAFR) (2000)
21. Li, Y., Gupta, K.: Motion planning of multiple agents in virtual environments on parallel architectures. In: ICRA, Roma, Italy, pp. 1009–1014 (2007)
22. Loscos, C., Marchal, D., Meyer, A.: Intuitive crowd behaviour in dense urban environments using local laws. In: Theory and Practice of Computer Graphics (TPCG 2003), pp. 122–129 (2003)
23. MASSIVE (2006), http://www.massivesoftware.com
24. Musse, S.R., Thalmann, D.: A model of human crowd behavior: Group interrelationship and collision detection analysis. In: Computer Animation and Simulation, pp. 39–51 (1997)
25. Pelechano, N., Allbeck, J., Badler, N.: Controlling individual agents in high-density crowd simulation. In: Proc. of ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA) (2007)
26. Pelechano, N., O'Brien, K., Silverman, B., Badler, N.: Crowd simulation incorporating agent psychological models, roles and communication. In: First International Workshop on Crowd Simulation (2005)
27. Pettre, J., Laumond, J.-P., Thalmann, D.: A navigation graph for real-time crowd animation on multilayered and uneven terrain. In: First International Workshop on Crowd Simulation (2005)
28. Petty, S., Fraichard, T.: Safe motion planning in dynamic environments. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 3726–3731 (2005)
29. Quinlan, S., Khatib, O.: Elastic bands: Connecting path planning and control. In: Proc. of IEEE Conf. on Robotics and Automation (1993)
30. Reynolds, C.: Big fast crowds on ps3. In: Sandbox 2006: Proceedings of the 2006 ACM SIGGRAPH symposium on Videogames, pp. 113–121. ACM Press, New York (2006)

31. Reynolds, C.W.: Flocks, herds, and schools: A distributed behavioral model. In: SIGGRAPH 1987 Proceedings, vol. 21, pp. 25–34 (1987)
32. Schreckenberg, M., Sharma, S.D.: Pedestrian and Evacuation Dynamics. Springer, Heidelberg (2001)
33. Shao, W., Terzopoulos, D.: Autonomous pedestrians. In: SCA 2005: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 19–28. ACM Press, New York (2005)
34. Simeon, T., Leroy, S., Laumond, J.: Path coordination for multiple mobile robots: a geometric algorithm. In: Proc. of IJCAI (1999)
35. Stentz, A.: The focussed D* algorithm for real-time replanning. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) (1995)
36. Still, G.: Crowd Dynamics. PhD thesis, University of Warwik, UK, Ph.D. Thesis (2000)
37. Sud, A., Andersen, E., Curtis, S., Lin, M., Manocha, D.: Real-time path planning for virtual agents in dynamic environments. In: Proc. of IEEE VR, pp. 91–98 (2007)
38. Sud, A., Gayle, R., Andersen, E., Guy, S., Lin, M., Manocha, D.: Real-time navigation of independent agents using adaptive roadmaps. In: Proc. of ACM VRST (2007)
39. Sung, M., Gleicher, M., Chenney, S.: Scalable behaviors for crowd simulation. Computer Graphics Forum 3, 519–528 (2004)
40. Sung, M., Kovar, L., Gleicher, M.: Fast and accurate goal-directed motion synthesis for crowds. In: Proc. of SCA 2005, pp. 291–300 (2005)
41. Thalmann, D., O'Sullivan, C., Ciechomski, P., Dobbyn, S.: Populating Virtual Environments with Crowds. In: Eurographics 2006 Tutorial Notes (2006)
42. Treuille, A., Cooper, S., Popovic, Z.: Continuum crowds. In: Proc. of ACM SIGGRAPH, pp. 1160–1168 (2006)
43. Tu, X., Terzopoulos, D.: Artificial fishes: Physics, locomotion, perception, behavior. In: Glassner, A. (ed.) Proceedings of SIGGRAPH 1994, pp. 43–50 (1994)
44. van den Berg, J., Patil, S., Seawall, J., Manocha, D., Lin, M.: Interactive navigation of individual agents in crowded environments. In: Proc. of ACM Symposium on Interactive 3D Graphics and Games (2008)
45. Warren, C.W.: Multiple path coordination using artificial potential fields. In: Proc. of IEEE Conf. on Robotics and Automation, pp. 500–505 (1990)
46. Yang, Y., Brock, O.: Elastic roadmaps: Globally task-consistent motion for autonomous mobile manipulation. In: Proceedings of Robotics: Science and Systems (August 2006)
47. Zucker, M., Kuffner, J., Branicky, M.: Multipartite rrts for rapid replanning in dynamic environments. In: Proc. IEEE Int. Conf. on Robotics and Automation (2007)

Populate Your Game Scene

Julien Pettré

INRIA Rennes - Bretagne Atlantique
Campus de Beaulieu, 35042 Rennes, France
julien.pettre@inria.fr

Abstract. We describe a method for populating large virtual environments with crowds of virtual pedestrians. Pedestrians are distributed in the environment by giving them goal destinations to reach. In order to facilitate this population setup, we assign identical destinations to batch of pedestrians, resulting in navigation flows. In order to preserve individuality of pedestrians, navigation is planned with variety and each pedestrians follows its own unique trajectory. This specific navigation planning technique answers user's queries very rapidly. As a result, impact of the insertion of a new navigation flow in the environment is directly observable, and interactive design is achieved. Once the problem of design solved, we address the problem of simulating the virtual population navigation. A level-of-details strategy is used to achieve on-line real-time simulation of large crowds, formed by up to tens of thousands of pedestrians. We illustrate our method on the typical example of a virtual city, and discuss our approach.

1 Introduction

1.1 Motivations

Many video games action take place in large virtual environments such as buildings, districts, entire cities or outer landscape. Players and virtual opponents are present in these environments, however, if the environment remains only populated with players, real or virtual, the scene may still look empty. Scene liveliness can be improved by filling up the place with a population of virtual inhabitants. A large number of them may be required to achieve this goal, even more when scenes are large. As a result, some aspects of this problem are close to the one of crowd simulation. However, video games applications have specific needs: among them, the need for a *eye-believable population simulation* (that can be opposed to the need for realistic results, and that also eliminate some classes of crowd simulation techniques such as cell-automata); the need for *interactivity* between the player and the population, which induces the need for on-line simulation; and finally, the need for giving *control* on the population activity to game designers.

Our objective is to provide tools for creating and animating large virtual populations, in order to make large environments alive. Some questions need addressing: how to setup a crowd and get control on its activity? how to simulate efficiently a crowd on the performances point of view? In the next section, we

© Springer-Verlag Berlin Heidelberg 2008

give a very brief overview of existing techniques for crowd design, visualization, simulation and motion planning. However, in this article, we will specifically focus on populations of pedestrians with behavior restricted to navigation. We will give key-ideas for designing large navigating crowds from simple and few high-level directives, and describe a crowd simulator exploiting a level-of-details strategy in order to reach high performances.

1.2 Related Work

Crowd Visualization and Design. Crowd Visualization is a specific problem with various solutions [1], [2], [3]. Briefly, these solutions rely on two major concepts. The first one is the level-of-details concept which consists in representing entities on the screen with a varying complexity, depending on its visibility and distance to the spectator's point of view; the more the entity is viewable (close and centered in the screen), the more its representation is accurate, and inversely. The second one is the use of impostors. Impostors are 2D images of entities which are used in place of 3D models when displaying the entity on the screen, allowing important computation time saves. It is possible to use such simple representations while preserving variety for visual aspect of individuals. CrowdBrush is a user-friendly tool for Crowd Design [4]. This tool is mainly dedicated to give visual aspects to many individuals in an easy manner, but also allows the distribution in space of the crowd and the definition of the inhabitants activity. Using CrowdBrush, goal destinations can be assigned to individuals, however, the user is in charge of providing a collision free path. Compared to this useful tool, we provide a solution to compute automatically collision-free paths for batches of pedestrians.

Crowd Simulation. Simulating crowds first consists in computing trajectories for several entities while taking into account their interactions and presence of obstacles in the environment. We briefly present 3 general methods to address crowd simulation among others: rule-based systems, particle systems and cell-automaton methods. Reynolds' rule-based model is a popular approach, allowing to compute motions for coherent groups [5]. Each entity's motion is influenced by other entities in its vicinity resulting in a global motion that can be compared to flock of birds or fishes. Reynolds declined later his approach in [6] to propose a complete set of steering behaviors such as seeking, fleeing, pursuing, etc. Using Particles Systems is another popular manner to simulate crowds [7], interactions between particles may be modeled by forces such as in [8], or by using potential fields [9]. Finally cell-automaton techniques relies on a floor-field structure, a regular discretization of the navigable space looking like a grid. Each cell of this floor field may be occupied or free. Each time-step, the content of occupied cells has a probability to be transferred toward a neighbor cell, modeling people's motion.

Crowd Control. We intend to populate environments with crowds of pedestrians with goal-directed navigation; this problem is thus reduced to a navigation planning problem. An exhaustive description of existing motion planning algorithms can be found in [10]. Briefly, three major classes of solutions can be distinguished. Firsts are *reactive methods*: the key-principle of these techniques is to model goals as attractive and obstacles as repulsive potentials [11]; the moving entity follows the gradient of the resulting potential field in order to reach its goal. This kind of techniques suffers some local minimum deadlocks, however, they are simple to implement and practical to use. *Sampling-based* motion planning is the second major class of approaches. These methods explore partially the navigable space in order to capture interconnected collision-free positions and paths into a graph structure (commonly called a roadmap). The well-known PRM (Probabilistic Roadmaps Methods, [12]) and RRT (Rapidly-Exploring Random Tree [13]) methods explore the free space in a random manner. A the opposite, deterministic approaches are based on a complete exploration of the free space, however, they can use discrete representation or complete representation of this space, resulting respectively in partial or complete solutions. Deterministic approaches result in a data structure, generally a graph, capturing both the geometry and the topology of the free environment. Navigation planning queries are solved using graph search algorithms. Deterministic approaches were recently used to adress the crowd navigation problem in [14].

1.3 Overview

Our approach allows a designer to populate a scene by adding progressively navigation flows in the environment. Navigation flows are subsets of pedestrians navigating back and forth endlessly between two given locations. The designer controls the distribution of the population in the environment only by selecting these destinations, and the pedestrians groups size (i.e., the number of individuals composing the group). By answering very rapidly designers' queries, they can immediately see the impact of the insertion of a new navigation flow, and compose a population by trial and error. There is no theoretical limitation on the number of flows, as the simulation complexity is mainly dependent on the total number of pedestrians composing the whole population.

These objectives are reached by combining 3 major ingredients:

1. *Navigation Graphs*: they result from a specific environment analysis which is similar to a cell-decomposition technique. Navigation Graphs are computed at a preprocessing stage, and support future navigation planning queries.
2. *Navigation Flows*: as defined above, they are sets of navigating pedestrians with common destinations. However, we want to preserve individuality in trajectories followed by each pedestrian. For this reason, we developed a specific navigation planning technique which computes a variety of solution paths. Doing so, from a unique navigation planning query, we can generate a variety of solution paths.
3. *Crowd Simulator*: it exploits a Level-of-Detail strategy, which allows to dispatch the available computation resources at the best. As for crowd visualization techniques, the key-idea is to get high-quality simulation in front of the user's point of view whereas the quality is progressively lowered in far and invisible areas.

Next sections detail each of these three ingredients. We demonstrate the efficiency of our approach in the Results section.

2 Navigation Graphs

Navigation Graphs ($\mathcal{NG}$) is a graph structure capturing the topology and the geometry of the *navigable space* of a given environment. Navigable space is the flat part of an environment (flat enough to allow human walk, according to a user-defined threshold slope) free of any obstacle.

A $\mathcal{NG}$ vertex captures a portion of the navigable space with a vertical cylindrical shape, high enough to include a human (typical height is 2 meters). A $\mathcal{NG}$ edge models a connection between two navigable cylinders, meaning they have a common portion of navigable space. Geometrically, passages between navigable cylinders are rectangular gates (with size as great as possible) included in the intersection of the two connected cylinders.

An example of Navigation Graph is displayed in Fig. 1 for a simple 2-D environment: obstacles are the blue regions of the figure, while the navigable area is left blank. The Voronoï diagram of the navigable area is displayed as well. $\mathcal{NG}$ vertices are represented as green circles centered on the Voronoï diagram. Edges are represented as red gates between overlapping vertices.

$\mathcal{NG}$ are computed automatically from the environment geometry and few user-defined parameters, as described in [15], in 5 successive steps:

1. *environment geometry sampling*: we create a regular grid of points matching the environment surfaces. As multi-layered environments may be considered, a simple elevation is insufficient since several elevations may correspond to given horizontal coordinates.
2. *grid mesh*: two neighboring grid points are interconnected when the slope of the in-between space is beneath the user-defined maximum slope angle and free of obstacle. With this stage, we provide a mesh capturing the navigable space in a discrete manner.
3. *clearance map*: we compute the clearance for each grid point, i.e. the distance to the nearest obstacle or high-slope. Given the the grid mesh computation

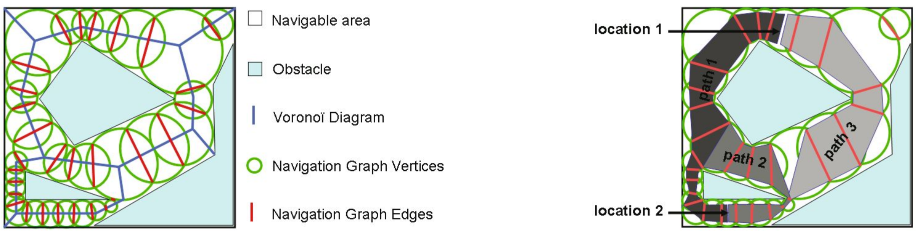


Fig. 1. *Left Image:* 2-D representation of a Navigation Graph. *Right Image:* Result of a Navigation Flow query: 3 Navigation Paths are found, each being a sequence of gates delimiting a corridor joining the 2 locations given as input of the query.

method, this distance is approximated by the distance to the nearest grid point not connected to all of its direct neighbors. Indeed, The lack of connection reveals the presence of an obstacle or a high-slope.

4. $\mathcal{NG}$ *deduction*: we then use a subset of grid points to compute the $\mathcal{NG}$ vertices. A graph vertex (cylinder) is created from a grid point by using it as centre and its corresponding clearance as radius. The grid point with maximum clearance is selected to create the graph vertex and all the grid points covered by the vertex are disabled for selection. The process is then reiterated until no more grid points remain for selection. Doing so, a majority of cylinders are centered on the Voronoï diagram corresponding to the environment.
5. *visibility pre-computation*: for each $\mathcal{NG}$ vertex, we compute the visibility of all other vertices according to the four main cardinal points. We can then use this information for visibility queries between moving entities.

On one hand, the use of an intermediate grid during computation makes the solution incomplete: navigable space is partially captured. But on the other hand, this discretization makes our method robust to complex and various environments.

3 Navigation Flows

A designer populates a virtual scene by creating as many *Navigation Flows* as desired. A Navigation Flow is a subset of pedestrians navigating endlessly between two locations of the environment. By choosing the number of pedestrians composing the flow as well as the locations to be joined, the designer is able to control the dispatching pedestrians in the scene. For example, the distribution of pedestrians into a virtual city is done according to the importance of some specific centers of interest, such as public buildings in a virtual city, shops, attractions, etc.

Collision free paths joining two locations can be computed easily using navigation graphs by simple graph search algorithms, such as A* or Dijkstra's. The search results in a *Navigation Path.* A solution path is composed by a sequence of connected edges to follow, i.e., a sequence of rectangular passages to cross. This sequence of passages delimits a solution corridor joining the two desired locations. The width of this corridor allow to: dispatch people, individualize their trajectories and let them avoid each other.

The individualization of navigation trajectories is important in order to preserve the believability of the obtained crowd motion. Indeed, if all pedestrians follow exactly the same path when navigating, strange queue formations appear as well as jams at some places in the scene with a negative impact on the resulting crowd motion. Following this idea, a single Navigation Path is even insufficient to drive a navigation flow. For that, we search for alternative paths that will finally compose Navigation Flows: a second level of variety in trajectories is thus obtained.

This search for alternative paths is illustrated in Fig. 1. A navigation flow is created between locations 1 and 2 as indicated on the right image of the figure. A first navigation path (the shortest) is found. As mentioned before, this navigation path is geometrically similar to a corridor joining the two locations. But, still in Fig. 1, one can see that paths 2 and 3 also compose the navigation flow. These two alternative paths will allow pedestrians to join their destinations passing by one side or another of obstacles in the environment.

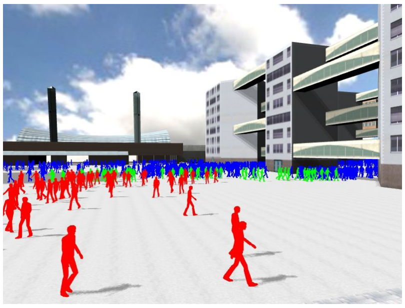
The existence of several navigation paths in a navigation flow is useful when jams appear during the simulation. The way pedestrians - belonging to an identical navigation flow - are dispatched inside a given solution path, and the way they are dispatched among the alternative navigation path is detailed in next Section. An extensive description of the Navigation Flows computation technique is given in [16].

4 Scalable Pedestrian Simulation

Once a navigation flow is created, simulation starts. During simulation, pedestrians' position is updated according to the path they are following and their position relatively to spectator's point of view. In this Section we describe:

1. How an individual trajectory is computed from the characteristics of the flow they belong to?
2. How the position-update accuracy is tuned according to the centrality and visibility of the corresponding pedestrian into the screen.

From Navigation Flow to Navigation Path. A pedestrian belonging to a navigation flow first chooses a navigation path among the available ones (i.e., one of the corridor as displayed in Fig. 1). This choice is made each time the pedestrian is at one of the extremity of the flow and have to go back (i.e., when at location 1 going to location 2 or inversely in Fig. 1). This choice is irrevocable



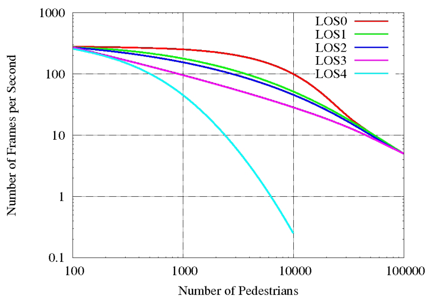


Fig. 2. *Left Image:* Distribution of different levels-of-details in scene. Most visible pedestrians (red) avoid each other, whereas far pedestrians (green and blue) do not avoid themselves and are updated at lower rates. *Right plot:* Performances obtained for each Level of Simulation LoS according to the number of pedestrians. LoS4, the most accurate, has quadratic complexity due to collision avoidance between pedestrians.

until he reaches his destination. Choice is based on the estimated travel time for each path belonging to the flow. Estimation takes into account the total length and density of population along the path, as described in [17]. Using a velocity /density statistical law $v = f(density)$, we can estimate the required time for crossing each area. By summing these crossing times, we compute the estimated travel time for each path. For example, a path composed by n edges has a travel time estimated to: $ETT_{path} = \sum_{i=0}^{n} ETT_{edge_i}$ with $ETT_{edge_i} = \frac{length_i}{v=f(density)}$. The local densities of population are not varying rapidly, as a result, the estimated travel times are updated at low frequencies. Finally, pedestrians choose the path currently having the lowest estimated travel time.

From Navigation Path to Individual Path. The distribution of pedestrians following a same navigation path is simply done according to an individual parameter p. This individual parameters (ranging from 0 to 1) determines a tendency of the pedestrian to walk on the left or the right of the followed corridor. More precisely, pedestrians follow way-points placed at each gate to cross. The position of the way-point WP within the gate AB is done as follow: $WP = A + p\boldsymbol{AB}$.

Levels of Simulation. Main objective of the simulation is to get believable results with high performances. High performances can be obtained by applying a strategy of levels-of-detail to the simulation. The principle is to focus the dedicated computation resources where it is the most needed, i.e., in the most visible parts of the environment. In areas at a far distance of the user point-of-view, or in the invisible parts of the environment, the simulation can be progressively simplified in order to save precious computation-time. Two major components in simulations allow to diminish the complexity of the simulation:

1. *update-rates:* Updating each individual at each simulation step results in a complexity which is linear with the number of simulated pedestrians. However, the spectator is able to see only a few part of pedestrians at a given instant with respect to his field of vision and the presence of obstacles occluding parts of the environment. We first improve the simulation performances by updating at high-frequencies positions of the most visible pedestrians while the other ones benefit from less frequent updates.
2. *reactive collision avoidance:* reactive collision avoidance between pedestrians moving in the same area is probably the most time consuming task in the simulation, with quadratic complexity. Collisions between pedestrians is required where the attention of the spectator is focused, however, elsewhere, collisions are not checked. Note that pedestrians are already dispatched in the environment using our specific Navigation Flow technique (both on several paths and inside each navigation path). As a result, at far distances (and obviously in invisible areas), the lack of collision-checks between pedestrians is not easily detectable.

Figure 2 illustrates our Level-of-Simulation (LOS) strategy. On the left image, the distribution of LOS is displayed: most visible pedestrians are in red. For them, collision avoidance is enabled and their situation is updated at each simulation step. Obtained performances are plotted on the right (LOS4 plot). For pedestrians who are far from the user point of view, the collision avoidance is first disabled (pedestrians in green on the left image, corresponding to LOS3 on the plot). Then, situation update rate is progressively lowered (LOS2, 1 and 0, pedestrians in blue in the image). In conclusion, Level-of-Simulations is a key principle in order to focus available computation resources where they are the most needed, and to improve drastically simulation performances: with low update rates and no collision avoidance, 10 000 pedestrians can be simulated at 100Hz.

5 Example and Discussion

In order to illustrate our method, we consider the following example of a virtual city displayed in Figure 3. This environment is large, 640 $\times$ 420 meters, and composed by approximately 100'000 triangles. A Navigation Graph is computed for this scene in 20 minutes approximately, composed by 5017 vertices and 8003 edges. Four visual output of the crowd simulation, we use the YAQ crowd visualization system developed at EPFL-VRlab.

In order to populate this environment, 7 navigations flows are sufficient. They join some main centers of interest into the city, such as an hotel, a train

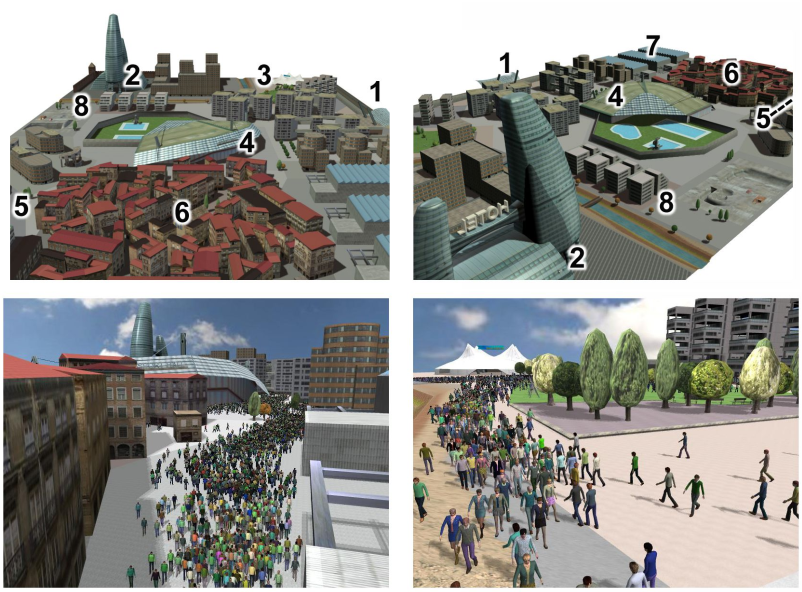


Fig. 3. *Top images:* Views of a virtual city, with 8 centers of interests corresponding to most important buildings (hotel, train station, church, circus, etc.). *Bottom images:* Views of the virtual population generated by creating several navigation flows between the centers of interest.

station, a circus, the old center of the city, etc. Each of the 7 Navigation Flow is composed by 5000 pedestrians, resulting in a virtual population made of 35'000 pedestrians. During Population Setup, each Navigation Flow is computed in less than a second, and as the simulation and visualization runs during this stage, we can evaluate immediately the impact of the insertion of new flows. Despite the small number of flows to populate this scene, all the places of this environment are occupied with pedestrians. This results from our specific dispatch method in Navigation Flow computations.

6 Conclusion

We presented a Navigation Graph based approach for populating scenes with crowds of pedestrians. The principle is to populate semi-automatically environments by planning the navigation of large crowds of pedestrians. Few high-level directives allow to populate large scenes with numerous pedestrians. We proposed efficient techniques in order to allow: first, interactive design of the population by answering quickly to user's navigation queries, and second, on-line simulation with real-time rates.

For efficient Navigation Planning, the key-idea is to search for varieties of solution. This allow to derive solution paths in order to individualize pedestrians' trajectories with no supplementary computation cost.

For efficient simulation of pedestrians, the key-idea is to use a level-of-details strategy in order to maintain a high simulation quality where it is the most needed, i.e., in front of the spectator, whereas the simulation quality is lowered elsewhere.

Our method fits the video-games field, and is applicable to the problem of populating large game scenes with large virtual population of pedestrians with low computation costs and believable results.

Acknowledgments

This work was realized in 2005-06 at EPFL-VRlab, directed by Pr. Daniel Thalmann. The project was funded by the European Research Community, by a Marie-Curie Action EIF 2005-mobility. The author would like to thank B. Yersin and J. Maïm who developed the crowd visualization system used in our examples, and M. Clavien and R. Krummenacher for environment design.

References

1. Aubel, A., Boulic, R., Thalmann, D.: Real-time display of virtual humans: Levels of details and impostors. IEEE Transactions on Circuits and Systems for Video Technology 10(2), 207–217 (2000)
2. Tecchia, F., Loscos, C., Chrysanthou, Y.: Visualizing crowds in real-time. Computer Graphics Forum 21(4), 753–765 (2002)

3. Dobbyn, S., Hamill, J., O'Conor, K., O'Sullivan, C.: Geopostors: A real-time geometry/ impostor crowd rendering system. In: I3D 2005: Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, pp. 95–102 (2005)
4. Ulicny, B., de Heras, P., Thalmann, D.: Crowdbrush: Interactive authoring of real-time crowd scenes. In: SCA 2004: Proceedings of the ACM SIGGRAPH/ Eurographics Symposium on Computer Animation, pp. 243–252 (2004)
5. Reynolds, C.W.: Flocks, herds, and schools: A distributed behavioral model. In: Stone, M.C. (ed.) Computer Graphics (SIGGRAPH 1987 Proceedings), vol. 21, pp. 25–34 (1987)
6. Reynolds, C.W.: Steering behaviors for autonomous characters. In: The proceedings of the 1999 Game Developers Conference (1999)
7. Bouvier, E., Guilloteau, P.: Crowd simulation in immersive space management. In: Proceedings of the Eurographics workshop on Virtual environments and scientific visualization 1996, London, UK, pp. 104–110. Springer, Heidelberg (1996)
8. Helbing, D., Molnár, P.: Social force model for pedestrian dynamics. Phys. Rev. E 51(5), 4282–4286 (1995)
9. Treuille, A., Cooper, S., Popovic, Z.: Continuum crowds. In: Proc. of ACM SIGGRAPH (2006)
10. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006), `http://planning.cs.uiuc.edu/`
11. Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. International Journal of Robotics Research 5(1), 90–98 (1986)
12. Kavraki, L., Svestka, P., Latombe, J.C., Overmars, M.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation 12, 566–580 (1996)
13. Kuffner, J.J., LaValle, S.: Rrt-connect: An efficient approach to single-query path planning. In: IEEE International Conference on Robotics and Automation, 2000. Proceedings. ICRA 2000, vol. 2, pp. 995–1001 (2000)
14. Sud, A., Andersen, E., Curtis, S., Lin, M., Manocha, D.: Real-time path planning in dynamic virtual environments using multiagent navigation graphs. IEEE Transactions on Visualization and Computer Graphics 14, 526–538 (2008)
15. Pettré, J., Laumond, J.P., Thalmann, D.: A navigation graph for real-time crowd animation on multilayered and uneven terrain. In: First International Workshop on Crowd Simulation (V-CROWDS 2005) (2005)
16. Pettré, J., Grillon, H., Thalmann, D.: Crowds of moving objects: Navigation planning and simulation. In: IEEE International Conference on Robotics and Automation, 2007, April 10-14, 2007, pp. 3062–3067 (2007)
17. Pettré, J., de Heras Ciechomski, P., Maïm, J., Yersin, B., Laumond, J.P., Thalmann, D.: Real-time navigating crowds: scalable simulation and rendering: Research articles. Comput. Animat. Virtual Worlds 17(3-4), 445–455 (2006)

Hierarchical Path Planning for Virtual Crowds

Kai Yip Wong[1] and Céline Loscos[2]

[1] University College London, Gower Street, London, WC1E 6BT, UK
[2] Universitat de Girona, Institut d'Informàtica i Aplicacions, Campus Montilivi, Girona 17071, Spain

Abstract. In this paper, we propose a hierarchical approach to path planning that scales well for large crowds. Crowds have become an important research field that presents challenges associated to the high complexity of potential behaviors. In most related work crowds are designed to follow global or local rules that infer intelligent navigation and behavior. Path planning methods can be applied in simulation where it is required to have an exact distribution of the crowd in the environment. However, the simulation is often created offline as path planning algorithms do not scale well with large numbers of agents, if each is considered individually. Our approach provides reliable paths that can be used for environment crowd distribution evaluations, and scales well with a high number of agents to run at interactive from rates for up to a few thousand agents.

1 Introduction

In the last ten years, crowds have become an important area of research. They present challenges in rendering [1,2,3], animation, and simulation [4]. Recent advances permit their use in the industry of entertainment (e.g., The lord of the ring trilogy) and video games (e.g., Ubisoft's Endwar), as well as emergency simulations [5,6] or city planning [7,8]. It is often identified that crowds need to:

1. *Navigate*, that is go from one point to another within the environment,
2. *Perform collision avoidance*, by detecting collisions against the objects of the environment and other agents,
3. *React* in function of the environment and the user with personalized actions,
4. *Perform intelligent actions* as part of the inherent animation characteristics of virtual humans.

In this paper we focus on the navigation part of individuals of a crowd. In most papers crowd navigation is inferred from a set of local rules accessed temporarily by the agents. This leads to realistic global navigation although agents are not fixed with any final goals. If one agent is to be followed it will usually end up strolling continuously in the environment. We propose a new hierarchical path planning approach to allow agents to follow a long term goal. Our objective is to provide a system for users who know where agents should go. For example, when simulating a big city, there will be a proportion of tourists that would

© Springer-Verlag Berlin Heidelberg 2008

follow certain paths that would lead to most tourist attractions, while agents representing residents would follow a path that would go from home to the public transports and to their working place. If one knows the proportion and distribution of the different possible goals, it is then possible to have a good estimation of a plausible simulation.

In this paper we provide a path planning algorithm that runs at interactive rates for up to a few thousands of agents. Typically such system could apply to city or public building planning and emergency simulation to simulate the average distribution in normal conditions and how long it would take to evacuate an area.

In the rest of the paper we first review in section 2 related work on crowd navigation and path planning algorithms while comparing them to our approach. The hierarchical path planning algorithm is presented in section 3. Results are presented in section 4 before we conclude in section 5.

2 Related Work

The work presented in this paper intersects with two different research areas: path planning and crowd navigation. Although several papers on crowd simulation already approach the navigation through some type of path planning, there is usually a significant difference that can be found in the literature due to the very different nature of the problem and the objectives to achieve.

Usually in path planning, one needs to find either a path or the optimal path. Precision and accuracy are often a requirement. The initial problem when considering how to plan a route or path through an environment is addressing how the environment is subdivided into manageable parts. A popular heuristic is the building up of a road map (or visibility graph) in an unexplored environment from the set of entire possible paths that a robot or agent can take; this probabilistic road map (PRM) is suitable for simple environments. Knowing a road map or graph, a depth-first search (DFS) algorithm can be used to find paths through the environment. The DFS is so called because the algorithm explores the connections of the most recently visited nodes before backtracking to previous nodes if and when unsuccessful. Djkstra and the a-star methods, focused cases of the DFS algorithm, are more commonly used for finding shortest paths. These algorithms are summarized in by Korf [9] in his survey of iterative search algorithms. Korf also includes navigation using potential fields as a method of guiding virtual agents. PRMs can be directly used for multi-agent path planning in an environment, as demonstrated by Sung et al. [10]. In this work they handle complex constraints (time, collision, target goals) for agent-to-environment and inter-agent interactions. Paths are calculated by solving a constraint-based motion graph for path segments merged and adjusted to meet common constraints.

Mamiya et al. [11] use potential fields created by an emission of waves to steer the agents that plot an optimal route through a potentially busy environment. A limitation of this approach is that this is only used to steer agents towards short term goals, in their case gates at a railway station, and is primarily concerned with navigating multiple agents in close proximity of many obstacles.

For crowd simulation, the agents that form the crowd need to find somehow their way and answer to a certain distribution. The developed algorithms are usually less accurate in the long term, answering often to short, local paths, while scaling for a large number of queries. Most of the methods on crowd navigation consider the trajectory of the agent moving through the environment or interacting with other agents in congested areas. They often differ from the structures used to store and retrieve information. Often there is no global navigation strategy such as for Kamphuis et al. [12] which base the method on agents' self-organization to avoid congestion, Tecchia et al. [2] and Loscos et al. [13] which base the navigation on local information, and Lerner et al. [14] which propose a navigation strategy learnt from video input. Longer term goals are taken in consideration by Lamarche et al. [15] for environment interaction, Wong et al. [6] for emergency evacuation, and Michael et al. [7] for density distribution.

From the approaches described above, we can conclude that there is not yet a method that can manage path planning scalable for multiple agents with different starting points and directions. In this paper, we propose a novel approach to path planning that makes use of a graph structure while approximating path calculations to directions. Our approach also differs in the sense that it is hierarchical, and therefore can provide different levels of detail for the path decision making algorithm.

3 Hierarchical Path Planning

3.1 Data Structures

The environment is represented by a set of data structures, all complementary. We consider that a road map of a real or virtual environment is available. Such a road map is illustrated in Fig. 1(a), in which the accessible areas are marked in white. This can be taken for example from GIS data or a traffic map of a real environment or an orthographic depth map of a rendered virtual environment [16]. The road map of the environment helps to create a *zone map*. Each zone represents a coherent area, which is either a junction or a portion of a road, neighbor to another area. In addition, zones locally describe accessible areas in the environment in order to indicate collision with the environment. An example of a zone map is shown in Fig. 1(b).

From the road map of a real or virtual environment, we also extract a graph, called the *road graph* for which the branches are the roads and the nodes are the junctions. An example is shown in Fig. 1(c). Assuming that the environment can be represented as a 2D structure, we also create a subdivision of the environment using a quad-tree, for which the highest level contains the entire environment and the lowest is such that each cell contains at most x nodes of the road graph (x chosen by the user).

A simplified graph, called *hierarchical graph*, is created such as, at each level of the hierarchy, links are created between each cell of the quad-tree when they

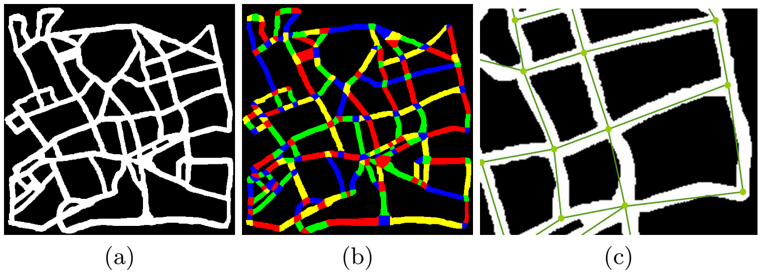


Fig. 1. (a) Example of road map extracted from a city street map. (b) Zones shown in different colors encoding the subdivided segments from the road map shown in (a). (c) An example of a graph shown in a zoomed in area of (a).

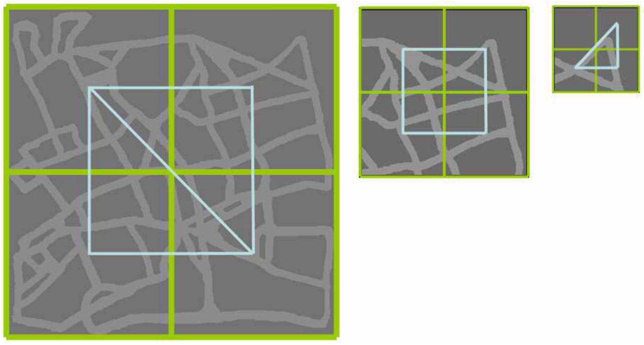


Fig. 2. Illustration of different levels of hierarchy of a quad-tree applied on an environment as shown in dark green. On the left hand side is shown the highest level, while the subsequent subdivision of the NE corner is shown on the right hand side. In light blue is shown the new graph structure built from the quad-tree, with cells linked only if there is at least one way to access directly one to another.

are directly accessible one to another. Such a hierarchical graph is illustrated in Fig. 2. Two neighbor cells are linked through their center only if there is at least one direct access through one road from one cell to another.

3.2 Decision Making

The strategy adopted by our approach is to refine the path selection when getting closer to the goal. The algorithm proceeds as follows, for each agent and using the same data structure for all, starting from the highest level of the hierarchy:

1. Identify to which quadrants belong the current position and the final goal of the considered agent.
   - Case 1. Both belong to the same leaf: go directly to step 6.
   - Case 2. Both are in the same quadrant: subdivide and iterate from step 1.
   - Case 3. They are in a different quadrant: go to step 2.

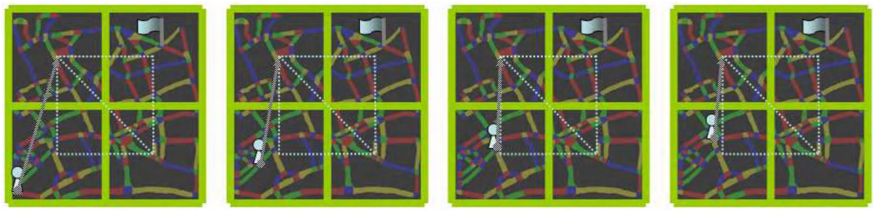


Fig. 3. Illustration of the decision making at a certain level of the hierarchy, with the sequence of events going from left to right. A path is chosen from the hierarchical graph to access from the starting point to the goal (shown here with a flag). At each change of zone, the agent checks for the overall direction to take by comparing its current position with the next target node.

2. Compute the path from the hierarchical graph at the given level in the hierarchy. Identify which node is the next logical node. This node must belong to a quadrant different to the one where the current position is. Calculate the direction between the centers of the current and target quadrant.
3. The direction calculated in step 2 is only indicative but helps adjusting a more appropriate direction adapted to the road map that will lead the agent from the current zone to the next one.
4. Navigate within the zone until a new zone or the final goal are reached. The direction set in step 3 does not exactly follow the road map. Instead the agent's local moving direction is adjusted to follow each road segment in a similar manner to [13,16]. In fact, any behavior strategy could be plugged in at this point.
5. A new zone has been reached. If the final goal is reached, stop. If the agent is still in the same quadrant, go to step 3. If the agent reaches a new quadrant, then go to step 1 and iterate.
6. Once in a leaf of the quad-tree, apply a traditional path planner (we used the Djkstra algorithm) from the current position to the final goal using the road graph.

The decision making at one level of the hierarchy (steps 2-4) is illustrated in Fig. 3. It is to be noted that the hierarchical graph helps agents calculating directions with a rough knowledge of road connections without the need of going down to the road graph structure. A simple distance-based direction calculation to the final goal wouldn't have worked as agents may have gone to an area not connected to their final goal and with the need of getting back.

4 Results

4.1 Algorithm Efficiency

The hierarchical algorithm was subjected to tests on timing, memory use, path accuracy and robustness on a test environment, that contained a variety of sharp and obtuse turns for the agents to navigate through.

Table 1. Timings in frame per second for the path planning only (no rendering), for 1 to 10,000 agents

Single agent (without GUI)	
Number of Agents	Iterations per second
1 agent	42,629,344fps
50 agents	169,467fps
100 agents	62,544fps
1,000 agents	6,696fps
10,000 agents	65fps

The algorithm was tested by running tests with increasing numbers of agents from 1 to 10,000 using random starting positions and goals. In order to measure the algorithm without the operating system diverting resources elsewhere, the tests were performed without the graphic interface and without agents checking for collision detection among themselves. However, collisions detection with the environment was conserved and included in the navigation strategy within the zones.

Timings results are shown in table 1. The time taken for all agents to reach their goals was used for the results. A series of 25 tests was used for the single agent and an average was taken. The table shows the algorithm to be robust with a population over 5,000, while the frame rate indicate real-time refreshment for up to the 10,000 agents tested.

One identified overload of our algorithm is that it requires to store several graphs. The overall memory usage is dictated by N, the number of levels used in the hierarchy, and since the structure is a quad-tree there are 3 possible connections that have to be stored at each level. The overall memory cost is given by: $\sum_{k=0}^{N} 4^N = \frac{1-4^N}{1-4} = \frac{4^N-1}{3}$.

4.2 Comparison with the a-Star Algorithm

The algorithm was compared with an a-star algorithm by choosing 25 random starting positions and goals. The simulation was run using both algorithms on each of the 25 paths using a single agent. The time taken to complete each run along with the number of iterations that were required to reach the goal was recorded. From this information an average count of the iterations performed each second or the frames per second could be calculated. The test was performed on a 1.8GHz machine running Windows XP. We measured an average of around 7 million fps for a single agent using an a-star against the 40 million fps for our hierarchical algorithm. The hierarchical algorithm has a much faster rate than the a-star. This can be attributed to the fact that the a-star calculates multiple possible paths and requires several storage structures to store and prioritize these possible routes. The rates are very high owing to the lack of a graphical interface.

The quality of the resulting paths from our algorithm was compared to the one of an a-star method using the same starting point and destination goals. Some of the examples can be seen in Fig. 4. If we were considering two tracked persons in an environment, both paths would be plausible.

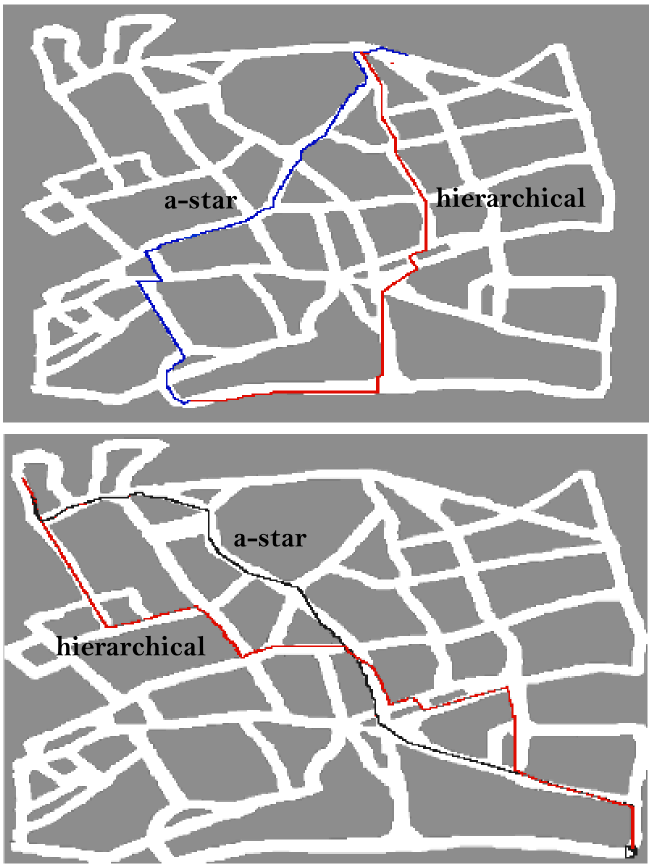


Fig. 4. Illustration of example paths produced by the hierarchical method (in red) compared to paths produced using an a-star (in blue)

5 Conclusions and Future Work

A scalable and robust system for virtual agent navigation is proposed. The algorithm runs at an interactive frame rate for a few thousand agents and the resulting generated paths can be compared to those produced by the a-star in finding a near-optimal solution without needing to store a large path.

In the future the algorithm can include more variety that will allow agents to take alternate routes that would get the simulation a more realistic looking appearance and appear to simulate everyday occurrences such as exploring or being lost. The is also the possibility of extending the algorithm use to scenarios such as evacuating an area or adapting to dynamic environments with only minor modifications. The algorithm can be expanded to take advantages of more adaptable storage structures than a quad-tree in cases where the environment is unevenly distributed.

References

1. Dobbyn, S., Hamill, J., O'Conor, K., O'Sullivan, C.: Geopostors: A real-time geometry/impostor crowd rendering system. In: Proceedings of the ACM SIGGRAPH 2005 Symposium on Interactive 3D Graphics and Games (SI3D), pp. 95–102. ACM Press, New York (2005)
2. Tecchia, F., Loscos, C., Chrysanthou, Y.: Visualizing crowds in real-time. Computer Graphics Forum 21(4), 753–765 (2002)
3. Ryder, G., Day, A.M.: Survey of real-time rendering techniques for crowds. Computer Graphics Forum 24(2), 203–215 (2005)
4. Thalmann, D., Musse, S.R.: Crowd Simulation. Springer, Heidelberg (2007)
5. Ulicny, B., Thalmann, D.: Crowd simulation for interactive virtual environments and VR training systems. In: Computer Animation and Simulation 2001, pp. 163–170 (2001)
6. Wong, K.Y., Thyvetil, M.A., Machaira, A., Loscos, C.: Density distribution and local behaviour for realistic crowd simulation. In: IADIS International Computer Graphics and Visualization (2007)
7. Michael, D., Chrisanthou, Y.: Automatic high level avatar guidance based on the affordance of movement. In: European Association for Computer Graphics, pp. 87–98 (2003)
8. Penn, A., Turner, A.: Space syntax based agent models. Pedestrian and Evacuation Dynamics, 99–114 (2001)
9. Korf, R.E.: Space-efficient search algorithms. ACM Comput. Surv. 27(3), 337–339 (1995)
10. Sung, M., Kovar, L., Gleicher, M.: Fast and accurate goal-directed motion synthesis for crowds. In: SCA 2005: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 291–300. ACM Press, New York (2005)
11. Mamiya, H., Kubo, M., Satoh, H., Namatame, A.: Agents on wave field: Simulation of traffic on train station. In: Industrial Electronics Society, 2000. IECON 2000. 26th Annual Conference of the IEEE, vol. 4 (2000)
12. Kamphuis, A., Overmars, M.H.: Finding paths for coherent groups using clearance. In: Pai, D.K., Boulic, R. (eds.) Eurographics/SIGGRAPH Symposium on Computer Animation, Grenoble, France, Eurographics Association, pp. 19–28 (2004)
13. Loscos, C., Marchal, D., Meyer, A.: Intuitive crowd behaviour in dense urban environments using local laws. In: TPCG 2003 (2003)
14. Lerner, A., Chrysanthou, Y., Lischinski, D.: Crowds by example. Computer Graphics Forum (Proceedings of Eurographics) 26(3) (2007)
15. Lamarche, F., Donikian, S.: Crowd of virtual humans: a new approach for real time navigation in complex and structured environments. Computer Graphics Forum 23(3), 509–518 (2004)
16. Tecchia, F., Loscos, C., Conroy, R., Chrysanthou, Y.: Agent behaviour simulator (abs); a platform for urban behaviour development. GTEC (2001)

Towards Embodied and Situated Virtual Humans

Stéphane Donikian[1] and Sébastien Paris[2]

[1] IRISA, Campus de Beaulieu, 35042 Rennes cedex - France
donikian@irisa.fr
[2] Laval University, Department of Computer Science and Software Engineering, Quebec, Canada
parissebastien@free.fr

Abstract. The main objective of the Bunraku project-team is to develop cross fertilization of researches in the fields of virtual reality and virtual human. Our challenge is to allow real and virtual humans to naturally interact in a shared virtual environment. This objective is very ambitious as it requires developing and federating several research fields. However, it is fundamental for several areas such as taking into account the human activity in manufacturing, the individual or collective training process, or the human study in cognitive sciences. In this paper, we will focus on the research done so far in the team that are concerning autonomous virtual humans.

1 Introduction

Virtual Human has at least two usages in a virtual reality environment, the first one is to represent a real human in the virtual world and it is then called an avatar, and the second one is to live autonomously inside the virtual environment, that is to say, perceive, decide and act on its own. Both uses require a geometrical representation of the human, but in the second case it is also necessary to model the behavioural activity instead of reproducing the activity of a real character. Modelling the human behaviour requests to take into account a certain number of topics such as understanding mechanisms underlying functions such as natural language production and perception, memory activity, multi-sensory perception, muscular control and last but not least the production of emotions. In short, it is necessary to be interested in the operation of various faculties that constitute together the human spirit, without forgetting their relation with the body. In complement of the study of these general mechanisms underlying any human behaviour, the work should also concern the study of human faculties in dedicated activities such as navigating in a city, using a work instrument or conducting a structured interview. The comprehension of the human behaviours requires competence in fields as varied as neurosciences, psychology or behavioural biology. Two types of approaches can be distinguished. The first one, known as the symbolic approach, consists in modelling the human behaviour in an abstract way in the form of modules describing each one a

© Springer-Verlag Berlin Heidelberg 2008

mechanism and relations of sequencialism or existing parallelism between them. It seeks to describe the mental processes by using symbols, judgements and mechanisms of logical inference. The second approach, known as systemic, consists to look inside the cerebral activity of patients subjected to various stimuli, according to well defined operational protocols. It is focusing more on concepts of signal transmission in networks, control and state feedback. The two approaches have different advantages: the first makes it possible to be abstracted from the biophysics processes present within the brain and to propose a modelling of the behaviour based on competence, while the second approaches, nearer to the neuro-physiological data, will be adapted to the modelling of the neuronal and sensori-motor activities. None of the models proposed in the two approaches is completely satisfactory to model the human behaviour in its whole. Indeed, our problem is not to reproduce the human intelligence but to propose an architecture making it possible to model credible behaviours of anthropomorphic virtual actors evolving/moving in real time in virtual worlds. The latter can represent particular situations studied by psychologists or correspond to an imaginary universe described by a scenario writer. However, the proposed architecture should mimic all the human intellectual and physical functions.

The important bibliographical study made during last years in the field of cognitive sciences pointed out the diversity, even the antagonism, of the approaches and the absence of federator models to propose an architecture allowing to connect together the various functions used in the human behaviour even for the simplest. The various approaches generally focus on one or the other of the functions or on a particular method of their relation. No theory exists for determining either the necessary or sufficient structures needed to support particular capabilities and certainly not to support general intelligence. There is however a general agreement on the decomposition into several layers (cf figure 1) or bands going from very low level control loops (reactive level), providing very fast responses to stimuli (sensory-motor control), to higher levels such as the cognitive one manipulating and reasoning on symbols, and the social one including personality,

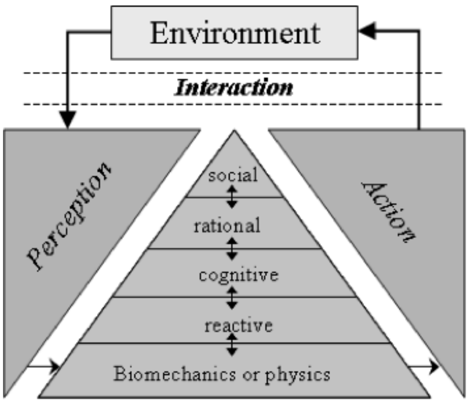


Fig. 1. The behavioural pyramid adapted from A. Newell[15].

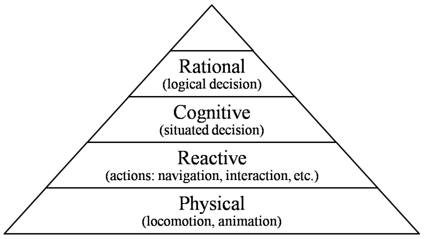


Fig. 2. Our multilayered model of decision

emotions and relation between humans. The reactive layer does not need to explicitly manipulate an abstract representation of the world, while the cognitive layer manages the abstract knowledge representation system. One of the main difficulties in the existing models is that usually symbols manipulated at the upper levels are not grounded into the world in which the lower level is making the virtual human react to its environment. The symbol grounding problem and the Chinese room problem are well known in artificial intelligence. We are proposing the first cognitive architecture proposing a full bidirectional link between four of the five layers: biomechanical, reactive, cognitive and rational (cf figure 2). Each layer exchanges specific information with the directly upper layer to inform it of some imposed constraints, and also with the directly lower layer to control it. Thereby, each layer is independent from the others, only requiring a set of identified data to work. Let us present now those layers, starting with the lower one.

2 Biomechanical Layer

The objective of this layer is to compute believable movements for human-like characters in real-time and interactive environments. Controlling the motion of virtual humans requires addressing four key points. The first one focuses on post-processing of raw data provided by motion capture systems [9] captured on a unique skeleton whereas a target skeleton will generally differ from this original one. The second point [8] consists in retargeting motion by using morphological matching and adaptation to the environment. It involves the solving of constraints in space and time but it does not meet the interactivity requirements. A third solution [2] is based on a frame by frame inverse kinematics approach, that does not guarantee a real-time animation for a high number of degrees of freedom or a high number of tasks to be respected. However, this solution allows the use of biomechanical knowledge when adapting the motion. The last point consists in using a mechanical model based on dynamical laws of motion. Once basic actions are available, it is possible to combine them in order to solve more complex tasks. For example, motion planning requires interpolating various kinds of gaits [20] or generating footsteps in order to drive a character from an initial to a goal configuration.

We have proposed methods to synchronize, merge and blend several captured motions (see figure 3)[13]. Each motion is seen as an action, which provides a

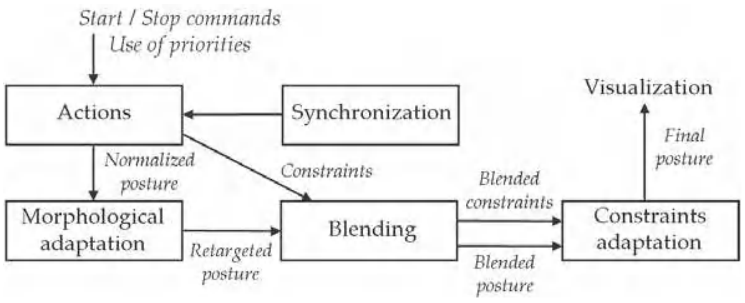


Fig. 3. General Architecture of our Animation Engine

succession of postures and constraint equations. The role of the synchronisation module is to perform dynamic time warping if actions are not compatible together. The system is then able to make the character step in environments, and move around without any a priori knowledge and without any pre-computation of possible transitions. As the target character is generally different from the original one, we have proposed motion retargeting methods based on normalised representations of the skeleton [10]. An efficient inverse kinematics and kinetics solver based on this specific representation of the skeleton solves the geometric constraints, such as the contact constraint of the feet on the ground. It enables animating more than 500 characters in real time (30Hz with a commodity PC). Moreover, the morphology can also modify the gesture, which requires designing a function linking the motion to morphology. This is achieved by interpolating gestures, which belongs to a database indexed by morphological parameters, such as limb lengths and proportion between segment lengths [21]. This method was applied to the simulation of locomotion for several different hominids. Whatever the used kinematical method, ignoring dynamics may lead to some artefacts. Dynamical parameters (mass and inertia), when taken into account, lead to more realistic motions. However, it entails more complex computations and related models are more difficult to control. Although we have introduced those parameters in secondary tasks of inverse kinematics offering more plausible motions, but it cannot generate a motion that always respects external forces. One of the main problems that still remain consists in understanding and modelling the laws that capture the naturalness of human motion in order to simulate realistic gestures.

3 Reactive Layer

Autonomous virtual humans are able to perceive their environment, to communicate with others and to perform various activities in accordance with the nature of the environment and with their intentions. Different approaches have been studied for the decision part of the reactive layer: sensor-effector or neural networks, behaviour rules, finite automaton. However, none of these approaches were sufficient to handle concurrency and time management in behaviours. The various approaches can be shared within a dataprocessing model implementing a

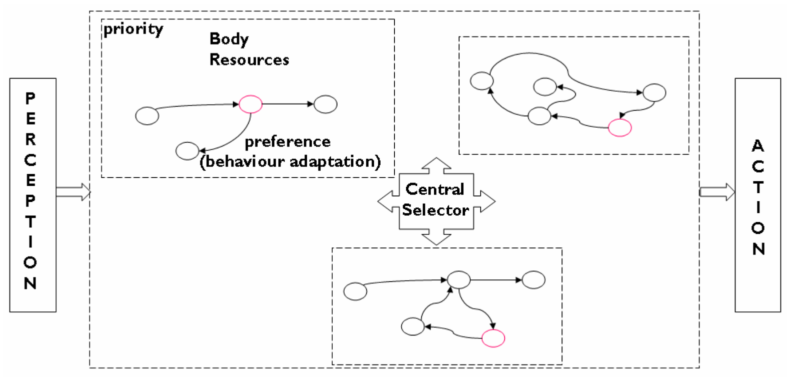


Fig. 4. Automatic coordination of reactive beahviours

hierarchy of "perception-decision-action" loops in which the decisionmaking process is real-time, and made up of concurrent and hierarchical state-machines having different scales of time. We have proposed the HPTS (Hierarchical Parallel Transition Systems) model [6]. It is composed of a language allowing the description of reactive behaviours through finite state machines and a runtime environment handling parallel state machine execution and offering synchronization facilities. States and transition can be redefined in the inheritance hierarchy and the state machines can be augmented with new states and transitions. The compilation phase translates a state machine in a C++ class that can be compiled separately and linked through static or dynamic libraries. The runtime kernel handles parallel state machine execution and provides synchronization facilities.

V. Decugis and J. Ferber [5] combine reactivity and planning capabilities in a real-time application, by extending the Action Selection Mechanism (ASM) proposed by P. Maes [12] into hierarchical ASMs. In the contrary of the ASM approach, W.J. Clancey [3] is defending that an activity does not have to be interrupted when another one requests to be performed. Instead, he defends the notion of competitive activation mechanism. In the above mentioned models, it is relatively complex to reproduce this type of interlaced behaviours, due to their mutual inhibition. We have proposed a new approach whose objective was to launch the required behaviours and let them arrange automatically according to the circumstances, the desires and the physical constraints of the agent [11]. Three new concepts were introduced within our reactive model: resources used at each step of behaviour, behaviour priority upon time, and degrees of preference allowing the adaptation of each behaviour to its context. A scheduler or central selector (cf figure 4) is trying, as well as possible, to respect the allowance of the resources according to the current priority of each behaviour. The description of a new behaviour, thanks to the mechanism of deadlock avoidance, does not require the knowledge of other already described behaviours. In this model, known as HPTS++, the behaviour coordination becomes thus generic and automatic.

4 Cognitive Layer

As humans are deliberative agents, purely reactive systems are not sufficient to describe their behaviour, and it is necessary to integrate the cognitive aspect of behaviour (beliefs and intentions) [7]. In order to do so, Badler et al. [1] propose to combine Sense-Control-Action loops with planners and PaT-Nets. In this system, like in others [16], the action is directly associated with each node, which doesn't allow the management of concurrency. Other planning systems elaborate plans by reasoning on actions and their associated resources [14]. However, such kind of approach is based on an omniscient point of view, where the agent can obtain any information from the informed environment to plan its path, which limits the realism of the simulation. Another approach, based on artificial vision, is using information retrieval from an image captured by a camera located on the head of the virtual human. This process, using the well known Zbuffer computes the perception of a virtual human. N. Courty [4] has extended this kind of approach by blurring the peripheral vision area in the perceived image and by introducing a saliency map calculation. Salient points are extracted from the perceived image and used in the visual attention guidance of a virtual human navigating without any goal in an urban environment. C. Peters et al. [19] use also an artificial vision model, but in their case it is coupled to a human memory model allowing to manage scanning and object retrieval techniques inside an apartment. A human being navigating in an environment is confronted to the problem of his own spatial localization. An approach is to consider that he or she is using a geocentric spatial cognitive map to navigate. Indeed, most of the cognitive works on that subject state that the knowledge a pedestrian can have of his environment differs a lot from what is really the environment: his perception and memory are most of the time incomplete and distorted [23]. Then it seemed interesting to us to study what should be added to the classical navigation architecture in behavioural animation, to simulate more realistic navigation behaviour. After a careful study of various works in the cognitive psychology field, the introduction of a spatial cognitive map and a human-like memory appeared necessary. The map is necessary to restrict the omniscience of the agent, and it allows the model to exhibit a more realistic navigation behaviour, due to the fact that planning is computed with incomplete knowledge of the environment. As the memory is dedicated to take into account a temporal factor in the simulation, the major consequence is that, paths taken to go from one location to another will not be the same at different moment during the simulation as the state of the agents memory is continuously evolving [22].

5 Rational Layer

This layer is in charge of the logical decisions of the autonomous agent. Decisions are based on un-embodied notions describing the goals and sub-goals of the agent at a conceptual level. This layer also organises these goals to find the feasible actions independently of their location. The logical decision of the autonomous

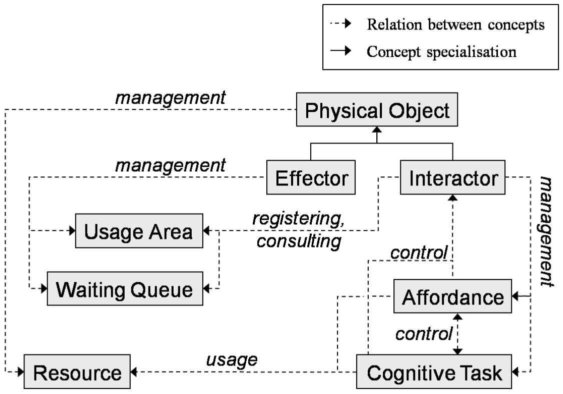
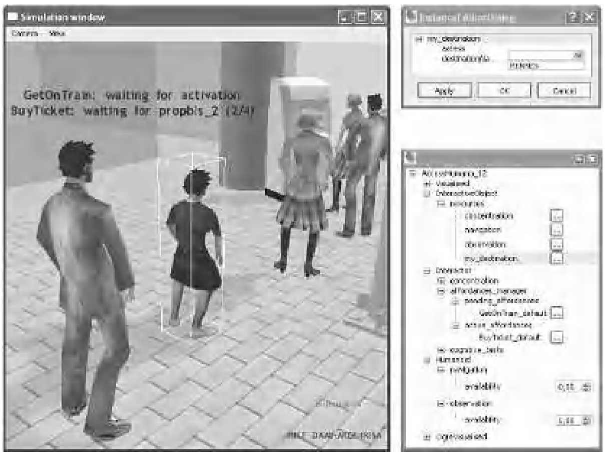


Fig. 5. Predefined set of concepts introduced within BIIO

agent is based on concepts describing its goals in the environment. These concepts are unified by an architecture which defines the way they operate together: BIIO for Behavioural Interactive and Introspective Objects. BIIO manages the concepts as objects in the sense of object oriented languages, i.e., by allowing the creation of a hierarchy of concepts specialising or unifying already existing ones. Another characteristic of BIIO is that any concept is endowed with introspective abilities, and thus able to communicate its components to others. This architecture also proposes a predefined set of concepts related to interaction behaviours. We define an interaction as any action between an autonomous agent and another embodied element situated in the environment, such as another autonomous agent or an equipment.

The set of concepts we introduce are defined in figure 5. *Physical Object* describes a situated object whose location is identified inside the environment. This concept is specialised in two other concepts: *Effector* and *Interactor*. An effector represents something which sustain an interaction, i.e. which participate to an interaction without having initiated it. represents an autonomous agent which can realise an interaction. An interactor is able to manage two concepts representing its behaviours: the *affordance* and the *cognitive task*. An affordance describes an interaction between an interactor and an effector, and so a behaviour of the interactor which is directly observable. These interactions are defined globally in the virtual environment, allowing to easily add new affordances without having to modify the effectors nor the interactors. Thereby, effectors and interactors are able to collect the affordances they are compatible with thanks to the introspective abilities provided by BIIO. Then, they have some generic processes to manage the collected affordances.

A cognitive task describes an internal behaviour of the interactor, which is not directly observable. A resource symbolises a property of a physical object, such as electricity for an equipment, or the ability to move for an autonomous agent. A resource can be transitory or permanent, and can have multiple internal properties which might evolve. Both behavioural concepts (affordance and cognitive task)



GetOnTram: waiting for activation
BuyTicket: waiting for propbis_2 (2/4)

Fig. 6. Some virtual humans waiting for the use of a vending machine

use resources to define their run, as for example to manage their competition. A usage area describes the positioning of an interactor in order to use an effector. A waiting queue describes the way the interactors must socially organise in order to wait for a specific interaction with an effector (cf figure 6). Waiting queues are managed on the same way than usage areas. In this approach, the situated decision is managed by a specialisation of the concept of cognitive task, *interact*, which is related to interactions. The goal of this specialised behaviour is to create a connection between the abstract decision of the interactor and its embodied abilities: perception, path planning[17], and navigation[18]. In fact, this cognitive task handles the competition between the interactors active affordances for the control of the agents physical abilities. *Interact* takes place as a hierarchical state machine, where each automaton has a specific role (cf figure 7).

The situated decision ability of our agents allows us to simulate complex behaviours with a really simple description phase: we just have to create the equipments (effectors) with their corresponding affordances, and then to assign an agent its final goal. For example, the behaviour of an outgoing passenger is shown in figure 8: the final goal assigned to the interactor is to reach a

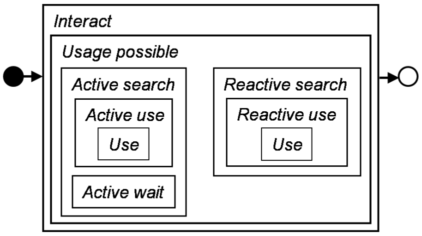
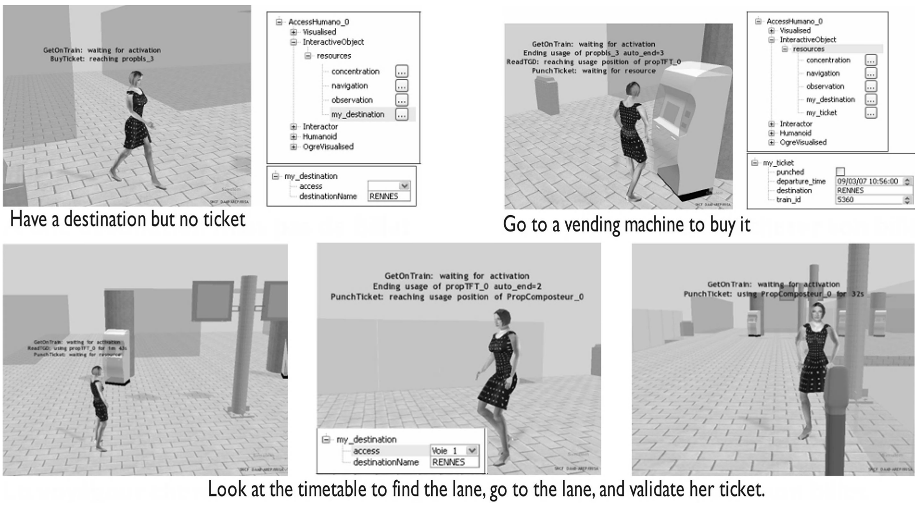


Fig. 7. The hierarchical state machine of *interact*



```
GetOnTrain: waiting for activation
BuyTicket: reaching propbls_3
```

```
AccessHumano_0
  Visualised
  InteractiveObject
    resources
      concentration
      navigation
      observation
      my_destination
  Interactor
  Humanoid
  OgreVisualised
```

```
my_destination
  access
  destinationName    RENNES
```

```
GetOnTrain: waiting for activation
Ending usage of propbls_3 auto_end=3
ReadTGD: reaching usage position of propTFT_0
PunchTicket: waiting for resource
```

```
AccessHumano_0
  Visualised
  InteractiveObject
    resources
      concentration
      navigation
      observation
      my_destination
      my_ticket
  Interactor
  Humanoid
  OgreVisualised
```

```
my_ticket
  punched
  departure_time     09/03/07 10:56:00
  destination        RENNES
  train_id           5360
```

```
GetOnTrain: waiting for activation
ReadTGD: using propTFT_0 for 1m 43s
PunchTicket: waiting for resource
```

```
GetOnTrain: waiting for activation
Ending usage of propTFT_0 auto_end=2
PunchTicket: reaching usage position of PropComposteur_0
```

```
my_destination
  access             Voie 1
  destinationName    RENNES
```

```
GetOnTrain: waiting for activation
PunchTicket: using PropComposteur_0 for 32s
```

Have a destination but no ticket
Go to a vending machine to buy it
Look at the timetable to find the lane, go to the lane, and validate her ticket.

Fig. 8. Example of the goal oriented behaviour of an outgoing passenger

destination by train, producing a set of sub-goals in order to buy a ticket (which creates a ticket resource added to the interactor), check the departure board, and punch the ticket (which modifies a property of the ticket resource); all of these interactions are performed with equipments which are chosen thanks to the interactors situated cognition ability.

Both models of decision presented in this paper, logical and situated, have been unified in a multilayer model of virtual agent (cf figure 9). The *interact* cognitive task manages embodied and situated behaviours, like navigation, with a purely logical process based on the abstract concept of affordance. This task can handle generically any affordance with a defined hierarchical state machine, and is able to take into account new affordance without being modified. A final problem needs addressing, which consists in choosing an effector inside the environment. This second part of the situated decision process is used by the active search automaton of the interact task. Its goal is to find the best couple of affordance and effector considering the relative priorities of the active affordances of the interactor, and the distance to travel to reach the effector. To do so, a path planning is performed inside the environment, whose ending condition is not a position, but an effector which is compatible with an active affordance.

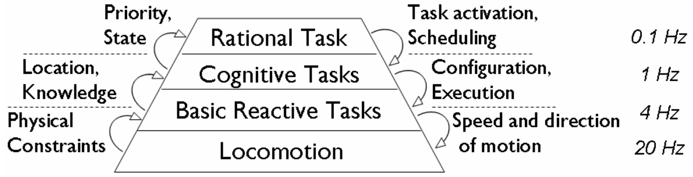
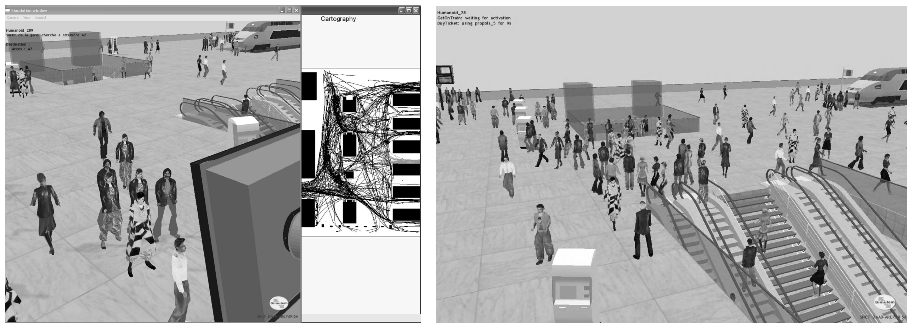


Fig. 9. A multilayered architecture of an embodied and situated virtual human



```
Humanoid_289
Sortir de la gare: cherche a atteindre A2

Destination :
 - Acces : A2
```

```
Humanoid_28
GetOnTrain: waiting for activation
BuyTicket: using propbls_5 for 9s
```

Fig. 10. User activities in a train station

The performances of the decision process are very good, with lesser than 0.5ms to take an initial decision, and with a negligible cost while only refreshing that decision (while walking towards a selected effector). Such high performances allow us to simulate one thousand of people in real time on a standard computer with full behavioural abilities, including the reactive and physical layers. This result is obtained with graphical output, which is responsible for approximately 50% of the total computation time. Left image of figure 10 is illustrating some characters waiting for the lane of their train in front of an information board. A window allows also to show trajectories followed by all pedestrians in the train station with the ability to discriminate by using a different colour those who will take a specific train. The right image illustrates the different activities of people inside the train station.

6 Conclusion

The principal key words of the whole work done so far on virtual humans are: how to model it, animate it, control it, and define its interactions with its environment and its congeneric. We wish in an immediate future to gather the whole of the work completed around the virtual human ranging from reactive to social behaviours and offering facilities for perception and action modelling. Those models have to take into account human behaviour characteristics in order to produce complex and believable movements and behaviours. Moreover, it has to respect the virtual reality constraints: real-time control and interactivity. To combine autonomy and believability, we are working on a unified architecture to model individual and collective human behaviours. This architecture should include reactive, cognitive, rational and social skills and manage individual and collective behaviours. This is one of our current challenges. We have proposed with BIIO a first answer to the symbol grounding problem by combining in a generic way cognition with a reactive embodied and situated human character. To tackle the embodiement of cognitive symbols we have developped a complex hierarchy of perception-decision-action loops, by managing the bidirectional exchange of information between the different levels. We will continue to develop this model by adding the social layer to the pyramid. We plan also to develop an audio-visual attentive coordination, integrating the management of human memory, the filtering of attention and the cognitive load. At the moment, motion control and behaviour model are usually seen as two independent software components, which does not allow a good solution for prediction and adaptation to the context. We will have to provide a better integrated solution, allowing the management of adaptable and expressive models of motion.

References

1. Badler, N., Reich, B., Webber, B.: Towards personalities for animated agents with reactive and planning behaviors. In: Petta, P., Trappl, R. (eds.) CPSA 1997. LNCS, vol. 1195, pp. 43–57. Springer, Heidelberg (1997)
2. Le Callenec, B., Boulic, R.: Interactive motion deformation with prioritized constraints. In: ACM/Eurographics Symposium on Computer Animation, Grenoble, France, August 2004, pp. 163–171 (2004)
3. Clancey, W.J.: Simulating activities: Relating motives, deliberation, and attentive coordination. Cognitive Systems Research 3(3), 471–499 (2002)
4. Courty, N.: Animation référencée vision: de la tâche au comportement. PhD thesis, INSA, Rennes, France (November 2002)
5. Decugis, V., Ferber, J.: Action selection in an autonomous agent with a hierarchical distributed reactive planning architecture. In: Sycara, K.P., Wooldridge, M. (eds.) Proceedings of the 2nd International Conference on Autonomous Agents (Agents 1998), pp. 354–361. ACM Press, New York (1998)
6. Donikian, S.: Hpts: A behaviour modelling language for autonomous agents. In: Fifth International Conference on Autonomous Agents, Montreal, Canada, pp. 401–408. ACM Press, New York (2001)
7. Funge, J., Tu, X., Terzopoulos, D.: Cognitive modeling: Knowledge, reasoning and planning for intelligent characters. In: Siggraph 1999, pp. 29–38. ACM Press, New York (1999)
8. Gleicher, M.: Retargetting motion to new characters. In: Cohen, M.F. (ed.) ACM SIGGRAPH, pp. 33–42 (1998)
9. Kovar, L., Gleicher, M., Pighin, F.: Motion graphs. ACM Transactions on Graphics 21(3), 473–482 (2002)
10. Kulpa, R., Multon, F., Arnaldi, B.: Morphology-independent representation of motions for interactive human-like animation. Computer Graphics forum 24(3) (2005)
11. Lamarche, F., Donikian, S.: Automatic orchestration of behaviours through the management of resources and priority levels. In: AAMAS 2000 and AAMAS 2002, pp. 1309–1316. ACM Press, New York (2002)
12. Maes, P.: Situated agents can have goals. In: Maes, P. (ed.) Designing Autonomous Agents, pp. 49–70. MIT Press, Cambridge (1990)
13. Ménardais, S., Multon, F., Kulpa, R., Arnaldi, B.: Motion blending for real-time animation while accounting for the environment. In: IEEE CGI (June 2004)
14. Nareyek, A.: A planning model for agents in dynamic and uncertain real-time environments. In: AIPS 1998 Workshop on Integrating Planning, Scheduling and Execution in Dynamic and Uncertain Environments, pp. 7–14 (1998)
15. Newell, A.: Unified theories of cognition. Harvard University Press, Cambridge (1990)

16. Noser, H., Thalmann, D.: Sensor based synthetic actors in a tennis game simulation. In: Computer Graphics International 1997, pp. 189–198. IEEE Computer Society Press, Los Alamitos (1997)
17. Paris, S., Donikian, S., Bonvalet, N.: Environmental abstraction and path planning techniques for realistic crowd simulation. Computer Animation and Virtual Worlds 17(3-4), 325–335 (2006)
18. Paris, S., Pettré, J., Donikian, S.: Pedestrian reactive navigation for crowd simulation: a predictive approach. Computer Graphics Forum 26(3), 665–674 (2007)
19. Peters, C., OŚullivan, C.: Synthetic vision and memory for autonomous virtual humans. Computer Graphics forum 21(4), 743–752 (2002)
20. Pettré, J., Laumond, J.-P., Siméon, T.: A 2-stages locomotion planner for digital actors. In: Proc. of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003), pp. 258–264 (2003)
21. Pronost, N., Dumont, G., Bérillon, G., Nicolas, G.: Morphological and stance interpolation in database for simulation of bipedalism of virtual humans. The Visual Computer 22(1) (2006)
22. Thomas, R., Donikian, S.: A spatial cognitive map and a human?like memory model dedicated to pedestrian navigation in virtual urban environments. In: Barkowsky, T., Knauff, M., Ligozat, G., Montello, D.R. (eds.) Spatial Cognition 2007. LNCS (LNAI), vol. 4387, pp. 421–438. Springer, Heidelberg (2007)
23. Tversky, B.: Structures of mental spaces. In: 3rd International Space Syntax Symposium, Atlanta, USA (2001)

Adaptive Body, Motion and Cloth

Nadia Magnenat-Thalmann, Etienne Lyard, Mustafa Kasap, and Pascal Volino

MIRALab, University of Geneva

Abstract. Virtual Try On (VTO) applications are still under development even if a few simplified applications start to be available. A true VTO should let the user specify its measurements, so that a realistic avatar can be generated. Also, the avatar should be animatable so that the worn cloth can be seen in motion. This later statement requires two technologies: motion adaptation and real-time cloth simulation. Both have been extensively studied during the past decade, and state of the art techniques may now enable the creation of a high quality VTO, allowing a user to virtually try on garments while shopping online. This paper reviews the pieces that should be put together to build such an application.

1 Introduction

Online shopping has grown exponentially during the past decade. More and more customers turn to this new medium, thus attracting the attention of the main industry players. The clothing industry is no exception to this rule, and various retailers now propose to purchase their garments online. A major problem related to the sale of garments through the internet is that the customer has no real way to try out a garment before purchasing it. Thus, a new kind of applications, called Virtual Try On, is now starting to emerge on the web and several companies [1] already offer this service.

However, due to the challenging nature of the task, the existing VTOs are quite simplistic. They do not allow to really assess the fitting of a cloth, but rather provide a visual idea of what the garment will look like, without animation. To move these applications to the next level, they should be able to accurately evaluate the fitting of a garment to a particular customer by taking into account its actual measurement, and also to provide animation so that the comfort can be estimated. To achieve these two goals, one first has to deform a body shape according to an input set of measurements. Second, animation clips should be adapted so that they match the newly created body shape. Eventually the cloth simulation should run in real-time, and be sufficiently accurate to make the obtained animation meaningful.

2 Human Body Modeling

3D human body models are a core element of computer graphics environments such as games, virtual modeling tools and virtual reality applications. The evolution of computer graphic techniques and hardware technology makes it possible to use more realistic models in these environments by also modeling the clothes, hair, gaze

© Springer-Verlag Berlin Heidelberg 2008

and motion. The primary characters of those environments, human body models, require specific techniques for real-time animation and realism. Because of their importance, specific research areas focused on cloth, face, hand, skin, anatomy and skinning to generate models which will improve the quality and realism of the virtual environments. Human body models which are subject to all those techniques and environments are initially generated by designers or acquired by 3D body scanners. These initial models are used as templates to generate a range of new ones through parametric deformations.

Parametric body modeling techniques can be categorized under the following titles: interactive, reconstructive, example based, anthropometric, and multi layered modeling. To achieve a higher degree of realism, a multi-layered approach is used to generate a body model that allows the interactive design of human bodies [2]. This approach requires intensive user interaction with slow production steps and limited number of control parameters. In case of reconstructive approaches, Lee et al. [3] used silhouette information from orthogonal images to deform a generic body model. Using face and body feature points along with front, back and side images, allows to construct an animatable human body using Dirichlet Free Form Deformation. Real-time deformations can be achieved by applying Free Form Deformation (FFD) techniques [4] (Fig. 1) while Zhaoqi et al. [5] used a contour based representation of the body model to generate skeleton-driven deformations.

With the new developments of hardware for computer graphics, the new trend was to use the GPU power for computationally intensive methods. Based on multi-resolution deformation on a high resolution meshes, Marinov [6] used the GPU for faster computation of the body deformations. Rhee et al. [7] parallelized the computation on GPU to exploit the efficiency of weighted pose space deformation. One of the recent techniques was proposed by Teran et al. [8] for modeling the muscles with finite element method to reproduce their physical behavior.

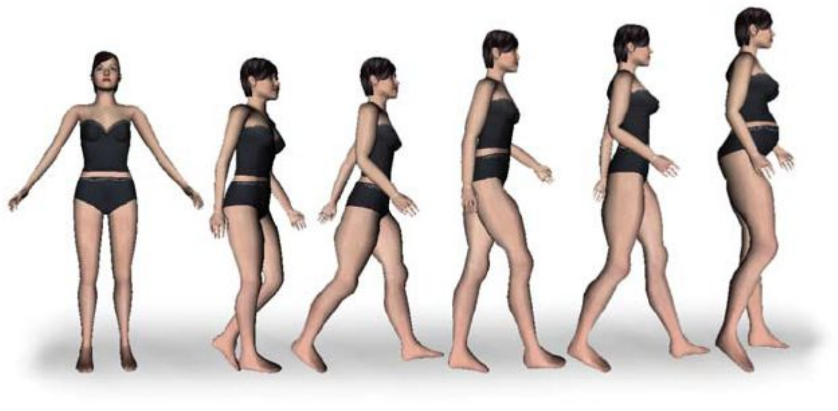


Fig. 1. Anthropometry based real-time body deformation and animation. From a template body model (left), different anthropometric regions (ankle, knee, thigh, height, front-back trunk, breast, arms) of the body are deformed while she is in walking locomotion

An early version of an approach based on examples was presented by Seo et al. [9]. They designed a template human body model with a set of manually defined landmarks on the mesh. According to user specified parameters that correspond to these landmarks, a scanned body model database is searched. The model that has the closest similarities to the specified parameters is fetched from the database. As the template model and the fetched one have the same set of predefined landmarks, the template body can be mapped on the other, so it can be deformed to have the same shape. The output of this process is a deformed template model with its adapted skinning information.

One of the other main body modeling techniques is to use anthropometry to segment the body model into logically deformable regions. International body measurement standards are available, defined under ISO-7250 and ISO-8559 [10] which is commonly used in anthropometry studies. These measurements are taken from the main body regions which are sufficient to reconstruct the similar body silhouette. These measurements are mainly used in clothing industry. For us, this is the easiest way for an end user to measure his/her body parts and enter his/her measurements as a parameter to the VTO application. Using a variation of the FFD technique and radial basis functions, predefined anthropometric body regions are deformed to generate the desired body shape. Deformation of the model surface is achieved in the direction of surface normals. This prevents the recalculation of the new surface normals, tangent space, and existing skinning weights. Deformation of overlapping body regions are obtained by blending the displacements on the intersecting regions. Continuity between the neighboring regions is preserved by using radial basis functions for producing smooth displacements on the boundaries. Since the anthropometry-based body regions are overlapping with the skeleton joints, the skeleton adaptation is achieved by re-scaling the bones by the same ratio as the region deformed over the joint. This approach makes it possible to dynamically deform the body model while it is being animated.

3 Motion Adaptation

Body deformers allow one to deform a body according to his/her requirements. Thus, the motion used for animation must be adapted to account for the changes applied to the character. For instance, if the character has to grab a surrounding object, then the movement of its arm must be changed so that he/she actually reaches the object. Similarly, self penetrations must be removed for avoiding collisions between the limbs and the body.

One of the most successful approach for modifying an existing animation is to use a spacetime optimization. It has been used by various works to address several aspects of the problem, such as foot plant enforcement, constraints compliance and physical properties of the motion. This approach modifies the entire motion clip in one pass, unlike Inverse Kinematics (IK) which deals with each frame individually (thus performing much faster) [11–13]. A motion clip seldom fits to the designers requirements. Thus, the researchers have focused their efforts in developing tools and methods for reducing the amount of work required to obtain the desired animation.

IK has proven to be efficient for self penetrations, but in case of deformable characters, the penetrations created by the growth of a limb tend to last for a long period of time. Using IK for handling these collisions creates discontinuities in the resulting motion and thus the spacetime approach seems well suited. The first work that used a spacetime optimization for adapting a motion was also from Gleicher [14, 15]. His method consists of defining characteristics of the motion that the user wants to enforce (e.g. feet on the ground, no sliding...) as a set of constraints that will be solved later on by a spacetime optimization. The optimization computes an adapted motion that re-establishes the constraints while preserving the characteristics of the original motion.

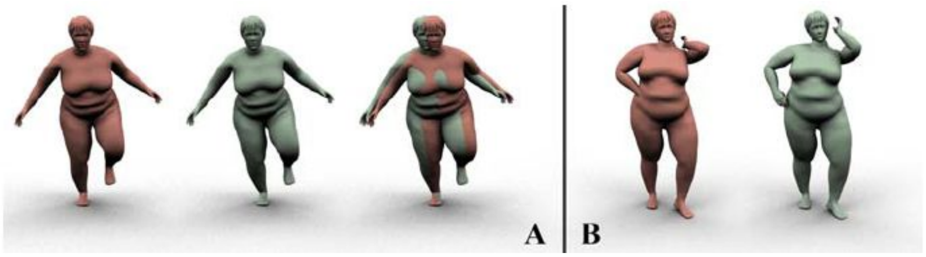


Fig. 2. Illustration of the motion adaptation process. The red figure is the original one, while the green is the adapted posture.

After a character is deformed, no self collision is allowed to take place and the movements Here we would like that no self collision remains while keeping the resulting motion close from the original, and physically plausible. Thus, the constraints to be satisfied become that the sum of all the penetrations remains zero. It may be that the penetrations occur at a difficult location, near the arm pit for instance. In this case, only one joint (the shoulder) can be modified to repair the motion. Any change applied to only one joint will be visible much as the entire kinematic chain attached to this joint will move along. One might be tempted to move the hand back towards its original location; however this is a bad idea because moving the hand towards the body will certainly create more self penetration. Thus, instead we propose to rotate the forearm towards it initial orientation. An example of penetration removal can be seen on figure 2B.

If the motion has changed much, then the balance of the character is not maintained any longer. Existing approaches can be used for addressing this issue, either by minimizing the energy consumption [16, 17] or by modifying the trajectory of the zero momentum point [18, 19]. As the animation clip being considered might not be optimal in terms of energy consumption, it does not make sense to use this criterion here. Moreover, previous approaches modified the ZMP trajectory on a per-frame basis and thus we proposed to modify the path of the ZMP using spacetime optimization [20]. In this implementation, we allow the character to lean so that the balance is maintained. Moreover, due to badly processed motion data, or in case of highly dynamic motions the ZMP does not remain within the supporting area. Thus instead of bringing it back under the character's foot sole, we bring it back *no further* than in the original motion clip. The result of such adaptation can be seen on figure 2A.

Eventually, footskate may have been created by the adaptation process and should be removed. Kovar et Al. [21] used a five steps kinematics algorithm to remove the foot skating. We developed a skin-based approach to perform the same cleanup [22] while Glardon et Al. [23] used a fully IK based approach. Figure 3 shows the effect of such process on a motion exhibiting foot skating.

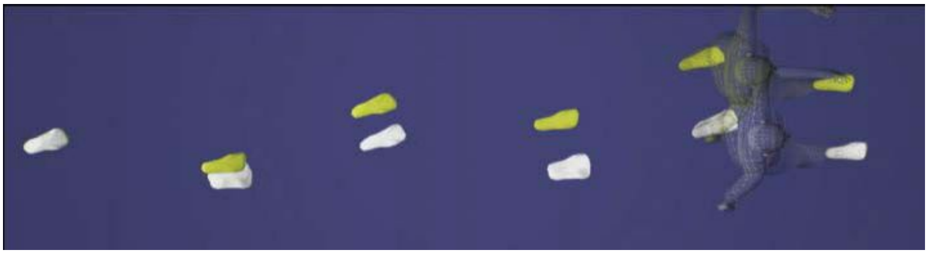


Fig. 3. Footprints left by a virtual character. The white prints exhibit foot sliding, while the yellow prints were left by the adapted animation.

4 Real-Time Garment Simulation

While simple computation models are able to simulate in real-time the animation of small fabric samples, simulation of virtual garments on animated virtual characters is a very time-consuming process. The biggest performance issues result from the complex and refined meshes necessary to describe the garment geometry in an accurate way. Mechanical simulation of such meshes requires a lot of computational resources to be dedicated to the mechanical evaluations of each mesh elements, along with the numerical integration of the resulting equations. Collision detection furthermore remains an important performance issue despite the use of sophisticated optimization algorithms, the virtual character itself being represented as a very complex geometrical object.

The still huge performance leap necessary for obtaining real-time simulation of complete garments cannot be obtained by further optimization of classic simulation techniques, despite the recent developments on simple models using particle systems, implicit integration and optimized collision detection. They require more drastic simplifications of the simulation process to be carried out, possibly at the expense of mechanical and geometrical accuracy. Among the possibilities are:

- Geometrical simplification of the mechanical description of the garment object, using rendering techniques (texturing, bump-mapping, smoothing) for reproducing small details (design features, smooth shapes, folds and wrinkles).
- Approximations in the collision interactions between the cloth and the body, for instance using approximate bounding volumes of force fields.
- Highly efficient mechanical models for cloth simulation, based on particle systems associated to efficient numerical integration methods which allow trading away dynamic accuracy to computation speed.
- Low-cost simplified geometric models for animating the cloth object, which approximate the mechanical behavior and motion of the cloth into predefined geometric deformations. Problems involve the design of adequate predefined motions that would represent the properties of the cloth in the many different mechanical contexts garments could be involved. These motions are usually

defined using analytic functions adapted to a particular context, or by automatic processes such as neural networks "learning" the cloth behavior from actual simulations [24, 25].

- Hybrid context-sensitive simulation frameworks which simplify the computation according to the current interaction context of garment regions, possibly mixing together rough mechanical simulation with small-scale specific simulation of features such as wrinkles [26].
- Integrated body-and-garment simulations where the cloth is defined directly as a processing of the skin, either as texture and bump-mapping (suitable for stretch cloth) or using local deformations reacting to the body motion using simplified mechanics [27].

All these techniques can be combined for designing a real-time system for garment animation, provided that the body animation system and the rendering pipeline is efficient enough so support these features with adequate frame rate.

4.1 An Example of Real-Time Garment Animation System

In this section, we intend to demonstrate the process of integrating the simulation of garments on an animated character in real-time. The main idea of this process is to create a unified representation of the body and the garment in a single object, by extrapolating the skinning information from the body surface to the garment surface. This extrapolated information would then be used either for animating the garment geometrically (through the same skinning deformation as the one used for animating the body), either to give to the mechanical simulator information for simplified collision detection between the cloth and the local body surfaces.

The automatic skinning extrapolation tracks the relevant features of the body shape ruling the animation of any vertex of the garment surface. This algorithm can be designed by extending a proximity map (nearest mesh feature algorithm) with additional visibility considerations for pinpointing the actual geometrical dependencies between the surfaces of the body and the cloth. A smooth blending between the weights of several nearest points smoothes the transitions between body parts. Additional smoothness criteria can also be also embedded so as to prevent any jaggy deformation over the garment surface. Further optimizations, such as the reduction of bone dependency count, should also be performed for reducing the computational time of skinning animation.

Collision data is obtained from the same nearest-feature algorithm used in the skinning extrapolation scheme, and optimized with specific distance and visibility considerations. Hence, for each vertex of the garment mesh, a set of vectors relating the nearest body features is stored. Then, during the animation, these vectors are skinned using the corresponding vertex weights, and collision distance and orientation can then be extracted from these vectors for adequate collision processing.

The mechanical engine is a fast and optimized implementation of the tensile cloth model [28]. It is associated to an implicit Backward-Euler integration scheme, which offers good performance along adequate robustness in this simulation context.

Through the use of an accurate mechanical model that offers good representation of the mechanical properties of cloth, it is possible to obtain a fairly realistic simulation which can suit the needs of fast fitting preview, during body motion, as well as body or garment resizing.

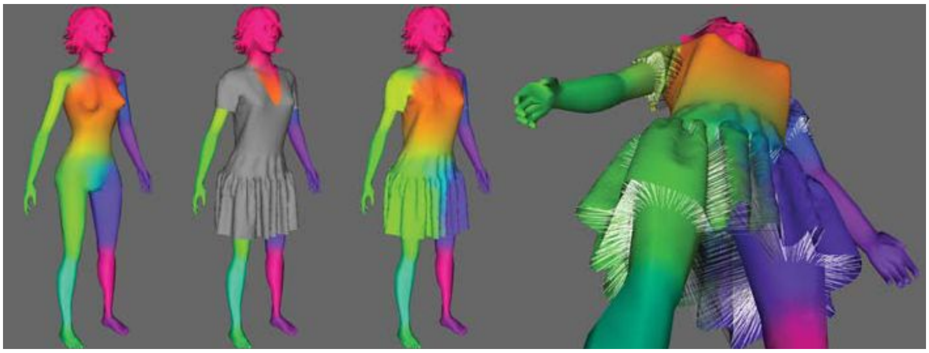


Fig. 4. The skinning weights of the mesh element (shown by the bone colors) (left) are extrapolated on the garment surface (middle) using through a smooth blending of the weights of the nearest mesh features. On the right, collision information is stored as vectors relating the orientation and distance of the potentially colliding body surfaces. These vectors are deformed by the skinning process during the body animation.

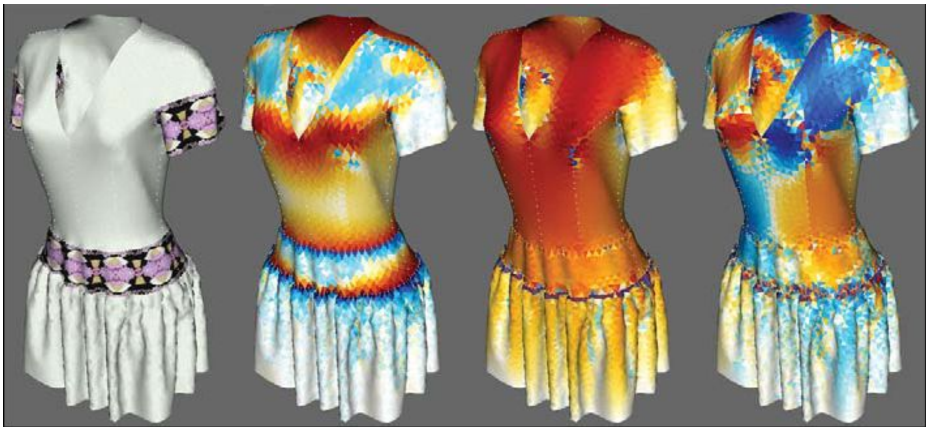


Fig. 5. Extraction of the weft, warp and shear tensile deformation values on the cloth surface using the 2D fabric surface coordinates of the patterns and the initial 3D shape of garment

5 Conclusion

In this paper we reviewed a set of techniques which can be used as the basic blocks needed to construct a VTO application. Parametric body modeling allow to deform a character in real-time, while motion adaptation ensures that the movements of the character will fit to the newly generated shape. To achieve real-time performances, tradeoffs were made regarding the accuracy of the cloth animation and meshes resolutions. The simplifications we made were specifically chosen because they allowed us to achieve the goals we had in mind. However, each application should make suitable simplifications so that the desired effects are still produced by the system.

Acknowledgements

This work has been developed under the European Project LEAPFROG IP (FP6-NMP-515810).We like to thank Christiane Luible, Nedjma Cadi and Bart Kevelham for their assistance in this work.

References

1. Redoute, L.: (2008) (accessed April 2008), `http://mannequin.redoute.fr/`
2. Nedel, L.P., Thalmann, D.: Modeling and deformation of the human body using an anatomically-based approach. In: Proc.Computer Animation 1998, pp. 34–40. IEEE Computer Society Press, Los Alamitos (1998)
3. Lee, W.-S., Gu, J., Magnenat-Thalmann, N.: Generating animatable 3D virtual humans from photographs. In: Gross, M., Hopgood, F.R.A. (eds.) Computer Graphics Forum (Eurographics 2000), vol. 19(3) (2000)
4. Kasap, M., Magnenat-Thalmann, N.: Parameterized human body model for real-time applications. In: Cyberworlds, 2007. CW 2007. International Conference on Cyberworlds, pp. 160–167 (2007)
5. Zhaoqi, W., Tianlu, M., Shihong, X.: A fast and handy method for skeletondriven body deformation. Comput. Entertain. 4(4), 6 (2006)
6. Marinov, M., Botsch, M., Kobbelt, L.: Gpu-based multiresolution deformation using approximate normal field reconstruction. Journal of graphics tools 12(1), 27–46 (2007)
7. Rhee, T., Lewis, J.P., Neumann, U.: Real-timeweighted pose-space deformation on the gpu. Computer Graphics Forum 25(3), 439–448 (2006)
8. Teran, J., Sifakis, E., Blemker, S.S., Ng-Thow-Hing, V., Lau, C., Fedkiw, R.: Creating and simulating skeletal muscle from the visible human data set. IEEE Transactions on Visualization and Computer Graphics 11(3), 317–328 (2005)
9. Seo, H., Magnenat-Thalmann, N.: An example-based approach to human body manipulation. Graph. Models 66(1), 1–23 (2004)
10. International organization for standardization (2008) (accessed April 2008), `http://www.iso.org/`
11. Choi, K.-J., Ko, H.-S.: Online motion retargeting. Journal of Visualization and Computer Animation 11(5), 223–235 (2000)
12. Baerlocher, P., Boulic, R.: An inverse kinematic architecture enforcing an arbitrary number of strict priority levels. The Visual Computer (2003)
13. Jeong, K., Lee, S.: Motion adaptation with self-intersection avoidance. In: Proceedings of the International workshop on human modeling and animation, pp. 77–85 (2000)
14. Gleicher, M.: Retargeting motion to new characters. In: Proceedings of SIGGRAPH 1998, Computer Graphics Proceedings, Annual Conference Series, pp. 33–42. ACM Press/ ACM SIGGRAPH, New York (1998)
15. Gleicher, M., Litwinowicz, P.: Constraint-based motion adaptation. The Journal of Visualization and Computer Animation 9(2), 65–94 (1998)

16. Popovic, Z., Witkin, A.: Physically based motion transformation. In: Proceedings of SIGGRAPH 1999, Computer Graphics Proceedings. Annual Conference Series. ACM Press/ACM SIGGRAPH, New York (1999)
17. Abe, Y., Liu, K., Popovic, Z.: Momentum-based parameterization of dynamic character motion. In: Proceedings of the ACM Symposium on computer Animation 2004. ACM Press, New York (2004)
18. Tak, S., Ko, H.-S.: A physically-based motion retargeting filter. ACM Transactions on Graphics 24(1) (2005)
19. Shin, H.J., Kovar, L., Gleicher, M.: Physical touch-up of human motions. In: Proceedings of the Pacific conference on computer graphics and applications. Wiley-IEEE Computer Society Press (2003)
20. Lyard, E., Magnenat-Thalmann, N.: Motion adaptation based on character shape. Comput. Animat. Virtual Worlds (to appear, 2008)
21. Kovar, L., Schreiner, J., Gleicher, M.: Footskate cleanup for motion capture editing. In: Proceedings of the ACM Symposium on computer Animation 2002. ACM Press, New York (2002)
22. Lyard, E., Magnenat-Thalmann, N.: A simple footskate removal method for virtual reality applications. Vis. Comput. 23(9), 689–695 (2007)
23. Glardon, P., Boulic, R., Thalmann, D.: Robust on-line adaptive footplant detection and enforcement for locomotion. Vis. Comput. 22(3), 194–209 (2006)
24. Grzeszczuk, R.: Fast neural network emulation and control of physics based models
25. Cordier, F., Magnenat-Thalmann, N.: A data-driven approach for real-time clothes simulation. In: PG 2004: Proceedings of the Computer Graphics and Applications, 12th Pacific Conference, Washington, DC, USA, pp. 257–266. IEEE Computer Society, Los Alamitos (2004)
26. Kang, Y., Choi, J., Cho, H., Lee, D., Park, C.: Real-time animation technique for flexible and thin objects (2000)
27. Cordier, F., Magnenat-Thalmann, N.: Real-time animation of dressed virtual humans. Comput. Graph. Forum 21(3) (2002)
28. Volino, P., Magnenat-Thalmann, N.: Accurate garment prototyping and simulation. Computer-Aided Design and Applications 2(5), 645–654 (2005)

From Motion Capture to Real-Time Character Animation

Franck Multon[1,2], Richard Kulpa[1], Ludovic Hoyet[2], and Taku Komura[3]

[1] M2S, University of Rennes 2, Av. Charles Tillon CS 24414, 35044 Rennes, France
[2] Bunraku, IRISA, Campus Universitaire de Beaulieu, 35042 Rennes, France
[3] IPAB JCMB, Edinburgh Univ, Kings Buildings, Mayfield Rd, Edinburgh EH9 3JZ, UK
{franck.multon,richard.kulpa}@uhb.fr, lhoyet@irisa.fr, tkomura@inf.ed.ac.uk

Abstract. This paper describes a framework for animating virtual characters in real-time environments thanks to motion capture data. In this paper, we mainly focus on the adaptation of motion capture data to the virtual skeleton and to its environment. To speed-up this real-time process we introduce a morphology-independent representation of motion. Based on this representation, we have redesigned the methods for inverse kinematics and kinetics so that our method can adapt the motion thanks to spacetime constraints, including a control of the center of mass position. If the resulting motion doesn't satisfy general mechanical laws (such as maintaining the angular momentum constant during aerial phases) the current pose is corrected. External additional forces can also be considered in the dynamic correction module so that the character automatically bend his hips when pushing heavy objects, for example. All this process is performed in real-time.

Keywords: Virtual human, real-time animation, kinematics, dynamics.

1 Introduction

Animating virtual humans thanks to motion capture data generally requires lots of precomputation and manual editing. Firstly motion capture data generally contain some holes and the Cartesian position of surface markers must be converted into joint angles. Most of the commercial softwares such as Motion Builder (product of Autodesk) and IQ for Vicon systems (product of Oxford Metrics) offer very efficient interfaces and algorithms to perform such processes. Secondly, motion capture data should be applied to the virtual character (called motion retargetting) that is used in the application leading to some corrections when the feet are not in contact with the ground during contact phases for example. Thanks to some research works in computer animation, this task can be performed almost automatically in a precomputation step. Then, new problems occur when this animation has to be applied to interactive environments such as video games. In the real-time engine, the 3D environment may be different from the one in which the actor performed his motion. The pose of the

© Springer-Verlag Berlin Heidelberg 2008

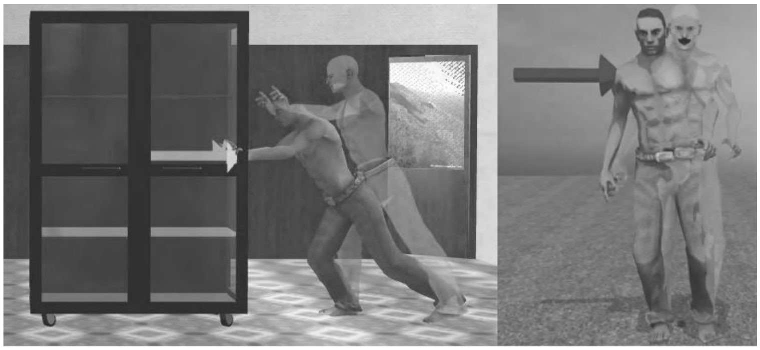


Fig. 1. Left) The motion of the character is adapted in real-time in order to push a heavy cupboard while the original motion was simply walking with the two hands placed in front of him (lightgray character). Right) The character reacts in real-time to an external force exerted on his shoulder while walking.

character should be adapted in order to take this new situation into account. This problem is generally solved concurrently to the motion retargetting step by precomputing some situations in advance. However, for a wide set of possible actions and situations, this manual approach leads to very long and fastidious work. Moreover the resluting database of motions is very large leading to long loading phases. It also requires a very large place on the storage support and memory.

Once motions are correctly precomputed, the real-time animation engine has to select the most convenient clip for a given situation and character. However, new behaviors and situations generally cannot be processed during real-time animation. Imagine an application where the user can create a character by tuning its dimensions and appearance. In that case, it's impossible to perform motion retargetting in advance. Let us consider now that this new character is driven in real-time by the user and has to interact with other players. For example, a user can put objects into a cupboard or other characters can concurrently interact with such a cupboard. The player wishes to see the virtual character adapting his pose as a consequence of a change of the mass of the cupboard (see figure 1).

This paper addresses this kind of problem by introducing real-time algorithms capable of adapting a motion clip to the morphology of the character, to kinematic constraints and to some dynamic constraints.

2 Related Works

Adapting motion capture data to new situations rises three main problems: solving kinematic constraints, ensuring continuity and preserving the original style. Displacement maps [1] [2] [3] where introduced in order to solve these problems. It consists in solving constraints when needed and then filtering the resulting discontinuous motion in order to obtain a continuous one. This approach can be used to retarget a motion to a character [4]. If the character that has to be animated has a different topology from the one that was used to process motion capture data (a character with an accurate model of the shoulder vs. an original model with a rigid torso for example), an intermediate skeleton can be used [5]. However, displacement maps require a complete knowledge of the sequence and the constraints in advance. In real-time interactions with a user, such constraints are not accurately known in advance as the user can modify the actions and the environment in an unpredictable way.

Real-time inverse kinematics [6] [7] [8] and kinetics [9] (ensuring the control of the position of the center of mass) can also be solved in real-time. In the specific case of foot-skate clean-up, this process is very fast [10]. For a whole-body pose control, it's very difficult to control more than one character in real-time whereas computation time used for this process could be used for processing other tasks in the animation engine. Moreover these techniques are only designed to solve kinematic and kinetic constraints in interactive time but cannot ensure that mechanical laws are satisfied (such as preserving the angular momentum constant during aerial phases or limiting joint torques).

To solve kinematic and dynamic constraints concurrently, several authors have proposed to optimize an objective function gathering all these constraints [11] [12] [13] [14] [15]. By nature, this process cannot be used in real-time animation as it requires a complete knowledge of the constraints in advance. Another solution consists in applying inverse dynamics and to tune to motion until the forces and torques reach acceptable values [16]. This method has been extended in order to deal with dynamic stability [17] and to correct the angular momentum in aerial phases thanks to time warping [18] or real-time local adjustments [19]. Dynamic filters were introduced in order to adapt an incorrect motion in order to satisfy the mechanical laws frame-by-frame [20]. It has been applied to obtain a stable motion from the dynamic point of view [21].

In order to react to external perturbations, it's also possible to simulate the passive behavior of the virtual human's mechanical system and to search a database of possible reactions the candidate that best fit the simulation conditions [22] [23]. This process calculates a whole sequence and is difficult to apply in real-time for interactive applications. Moreover, only local perturbations such as pushes are considered. It cannot be used to compensate continuous external actions, such as pushing an heavy object.

In this paper, we propose a framework to animate virtual humans thanks to motion capture data while taking kinematic and dynamic constraints into account. This framework was designed for real-time animation (per-frame solvers). Section 3 provides an overview of this framework. All this framework is based on a morphology-independent representation of motion that is described in section 4. Then, the inverse kinematics and kinetics solver (section 5) and the on-line dynamic correction (section 6) are presented.

3 Overview

The whole framework is able to synchronize and blend several motion capture data in order to solve a complex task in real-time, as described in [24]. When all the desired motions are blended into a unique pose, this latter can be adapted to a set of constraints. This paper focuses on this part of the framework only. Let us consider q the set of data that are associated to a pose of the character. In this paper, these data are based on a morphology-independent representation of motion, as described in section 4. At time t, q is associated to a set of kinematic and kinetic constraints (modeled as equality equations $f_i(q) = c_i$).

Before solving these constraints, q must be scaled to the virtual human's morphology, as depicted in figure 2. The resulting vector q_s is not made of joint angles any more but consists of Cartesian and angular data. The inverse kinematics and kinetics solver is applied on q_s in order to find a pose that satisfies a compromise of all the constraints. At this step, the solver delivers joint angles that are compatible with the virtual character. However, the resulting motion doesn't take external forces into account so that the angular and linear momentums may be incorrect from the mechanical point of view. The Dynamic Correction module aims at modifying the resulting angles in order to ensure that external forces are taken into account. This module doesn't compute the joint torques but is based on analyzing the mechanical constraints applied to the global system. Hence, during the aerial phase, the angular momentum should remain constant and the acceleration of the center of mass should equal gravity. During the contact phase, we assume that the character should counteract the external forces to preserve as much as possible the initial movement. To do so, the system can tune the orientation of groups of body segments. The goal is to modify the ground reaction force, the center of pressure and the position of the center of mass in order to counteract the new imposed external forces, as depicted in figure 1.

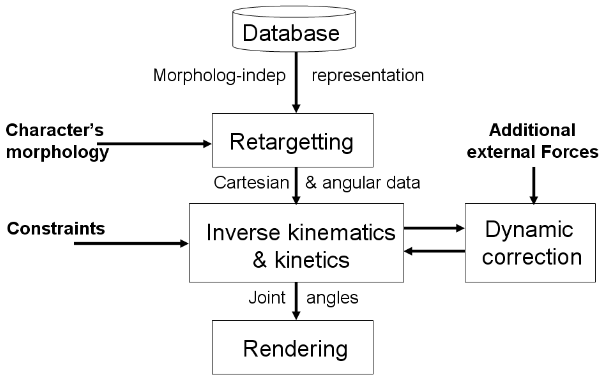


Fig. 2. Overview of the motion adaptation framework

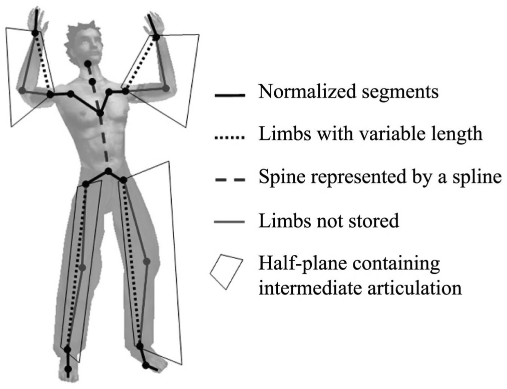

Normalized segments
Limbs with variable length
Spine represented by a spline
Limbs not stored
Half-plane containing intermediate articulation

Fig. 3. Morphology-independent representation of motion

4 Morphology-Independent Representation of Motion

The morphology-independent representation of motion was preliminary described in [25]. It's based on a Cartesian normalized representation of a posture. In this representation, the human body is subdivided into kinematic subchains (see figure 3) that describe parts of the skeleton. Those kinematic chains are divided into three main types:

– the normalized segments that consist of only one body segment (such as the hands, the feet, the clavicle and the scapula). They are normalized by their length ;
– the limbs with variable length that encode upper and lower limbs composed with two body segments each. In this representation, the intermediate joints (elbows and knees) are not encoded given that their positions are directly linked to the characters anthropometric properties. To retrieve them, an analytical solution derived from IKAN [26] is used ;
– and the spine is modeled by a spline (as suggested in biomechanics) which advantage is that it can be easily subdivided into as many segments as wishes, depending on the character description. This spline is easily normalized by scaling the coefficients between 0 (close to pelvis) and 1 (close to the skull).

Given this data structure, every motion capture data or edited trajectory (whatever the size of the initial skeleton) can be used as an initial pose. This pose is then adapted to the new skeleton by simply multiplying all the normalized data by the dimensions of the new character.

5 Inverse Kinematics and Kinetics

Once a set of parameters q_s is scaled to the size of the virtual character, the system has to solve the equality constraints $f_i(q_s) = c_i$. Generally, this task is performed by an inverse kinematics solver that consists in locally linearizing f_i to

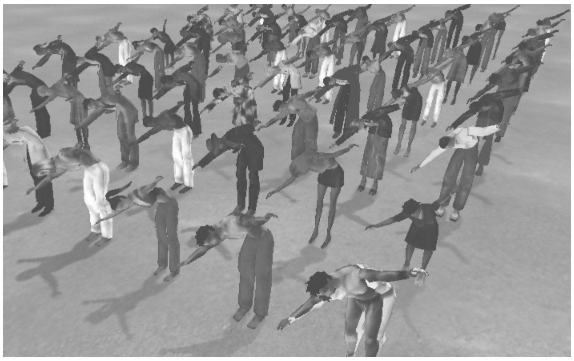


Fig. 4. Dozens of characters controlled through interactively constraining the wrists and ankles. The center of mass is also constrained over a vertical line going through the middle of the feet.

inverse it. However, the computation cost is very high and this process sometimes leads to unrealistic poses. The main problem is to solve constraints expressed in the Cartesian frame whereas q_s is a set of joint angles. With our representation based on relative Cartesian positions, the problem is simpler. Hence, for a group of body segment, it's possible to find an analytical solution for each constraint. For the arm, it's possible to determine very efficiently the new location of the wrist in the shoulder reference frame while preserving the initial orientation of the arm. The same kind of direct solution can be found for each group of body segments (limbs, trunk and head).

The main problem is to control the whole body whereas the solutions can be found locally for each group of segments. To solve this problem, we have adapted a Cyclic Coordinate Descent algorithm [27]. It consists in iteratively adapting the groups in a specific order until convergence. The order is selected to adapt the distal segments first because they are associated with less energy than the trunk or the whole body position and orientation. Moreover, the limbs offer a wide range of reaching positions that generally enables to solve the constraints in only one iterations (for targets (c) placed close to the body). The corresponding algorithm is:

```
it = 0
Do
  adaptGroup(HEAD)
  adaptGroup(RIGHT ARM)
  adaptGroup(LEFT ARM)
  adaptGroup(RIGHT LEG)
  adaptGroup(LEFT LEG)
  adaptGroup(TRUNK)
  adaptRoot()
  completed = computeError()
While ( (it++ < maxIt) & (not completed) )
```

This method also enables us to control the position of the center of mass [28] by offering immediate position of the local centers of mass for each group. Thus, analytical solutions can also be proposed for each body group to displace its local center of mass to a convenient place. However, for this process, the order in which the groups are adapted is different: from heaviest masses (leading to a large displacement of the global center of mass) to lightest ones (small displacements). This way, the minimum number of necessary groups is affected with this method. Figure 4 depicts dozens of various characters that have to satisfy the same geometric constraints (placing the two wrists and ankles onto targets) while ensuring that the center of mass stays along a vertical line.

6 Dynamic Correction

The dynamic corrections consist in adapting the joint angles if the corresponding whole-body motion doesn't satisfy the mechanical laws. It can be subdivided into two parts:

- during aerial phases, some global mechanical values are well known. For example, the angular momentum remains constant and the acceleration of the center of mass equals gravity. Let function $h(q_s)$ returns the angular momentum according to the joint angles. If an error Δh in the angular momentum appears, it's possible to retrieve the joint angles that eliminates this error by locally linearizing h (see [19] for details).
- during contact phases, the way the global mechanical values should change is more complex. If the user adds external forces that were not initially considered, the system should be able to adapt the sequence of poses to react to this perturbation. For example, when a character is pushing a cupboard, he should adapt his poses in order to react to the corresponding external force. The same way, when a character is turning, he should react to centrifugal accelerations by bending the whole body inside the turn.

Let us consider now the last item. The main goal is to perform the initial task as much as possible while reacting to the continuous external forces F_e added in real-time. To do so, the system can act with two complementary methods:

- displacing the center of pressure COP into the base of support in order to create a new torque $\Delta COP \times R$ where R is the ground reaction force. If the required COP remains in the base of support to counteract all the perturbation, no other change is needed; nothing is changed in the character's pose.
- if the first solution is not enough to counterbalance F_e, the next step consists in changing the ground reaction force R. Theoretically R becomes $R - F_e$. All the components of R change but the vector should stay within the cone of friction forces. Each change of the ground reaction force leads to a change of the global orientation and position of the center of mass. As the character is attached through kinematic constraints to the ground, it leads to a change of the pose of the character, as depicted in figure 1.

For the latter, the main problem is to express the Newton's laws as a function of the global orientation of the character q_1 at time t:

$$\mathcal{M}_W(q_1) + \mathcal{M}_R(q_1) + \mathcal{M}_{F_e} - \dot{H}(q_1) + \mathcal{M}(m\ddot{x}(q_1)) = 0 \tag{1}$$

$\mathcal{M}_W(q_1)$ is the torque due to the body weight, $\mathcal{M}_R(q_1)$ is the one due to the ground reaction force (linked to the position of the center of pressure), $\mathcal{M}_{F_e}$ is the one due to the additional external forces and $\dot{H}(q_1)$ is the derivative of the angular momentum. The center of pressure is assumed to be the Zero Moment Point, usually utilized for stabilizing biped robots [17]. Without additional external forces, when a motion is retargetted to a new character, this sum is certainly different from 0. This is due to a set of approximations: the model is different from a real human body, the body segments length and mass properties are also different. So equation 1 becomes:

$$min_{q_1} \Big(\mathcal{M}_W(q_1) + \mathcal{M}_R(q_1) + \mathcal{M}_{F_e} - \dot{H}(q_1) + \mathcal{M}(m\ddot{x}(q_1))\Big)^2 \tag{2}$$

This expression is minimized to calculate the global orientation of the character that best satisfies equation 1. When the global orientation is known, the kinematic constraints may be unsatisfied and a new inverse kinematics step is performed to locally correct the pose of the character, as depicted in figure 2. This minimization includes the perturbations due to the external forces. As a consequence, the global orientation is modified not only to compensate the change of morphology but also the addition of those external forces. q_1 should also remain in the neighborhood of the current pose in order to avoid discontinuities. Hence, the search space is quite small and a simple iterative method can find an optimal solution with only few steps.

Left part of figure 1 depicts a character that has to push a furniture. Depending on the weight of the furniture and the friction forces on the ground, the virtual human has to compensate an external force applied to his hands. In this figure, the lightgray character stands for the original pose, without external force. The other character is obtained after our method for a 50Kg furniture. We can see that the system automatically rotates the whole body to compensate the external forces exerted by the furniture. The kinematic constraints (connecting the feet without sliding on the ground and putting the hand on the furniture) are also satisfied. This correction is performed in real-time.

7 Conclusion

We have described a framework to control motion of virtual characters thanks to motion capture data while satisfying kinematic constraints with very few computation time. In addition to kinematics, the system is able to correct poses that doesn't satisfy general mechanical laws if we only consider the center of mass mechanical system. Although this approach is limited to the global system, it enables us to calculate motions that can take continuous external forces into account. It's the case when a character has to push or carry objects, for example.

The goal is to propose very fast computation instead of calculating all the joint torques that are necessary to control the character's mechanical skeleton. Although this very fast computation leads to some approximations, it can be used for animating numerous characters concurrently while taking some dynamic effects into account: bending the body while turning, climbing the trunk while walking on a leaning ground, hanging heavy objects...

In the future, this method should be extended to deal with local adjustments instead of global ones. Moreover, with the method presented in this paper, this character reacts instantaneously to external perturbations whereas some latency occur in real world. We should also take this latency into account, otherwise the character may look like superman in some cases.

Acknowledgements

This work has been supported by the "Conseil Régional de Bretagne", the "Conseil Général d'Ille et Vilaine" and Rennes Metropole.

References

1. Gleicher, M.: Motion editing with spacetime constraints. In: Proceedings of Symposium on Interactive 3D Graphics, pp. 139–148 (1997)
2. Lee, J., Shin, S.: A hierarchical approach to interactive motion editing for human-like figures. In: Proceedings of ACM SIGGRAPH 1999, pp. 39–48 (1999)
3. LeCallennec, B., Boulic, R.: Interactive motion deformation with prioritized constraints. In: Boulic, R., Pai, D.K. (eds.) Proceedings of ACM/Eurographics SCA, Grenoble, France, pp. 163–171 (August 2004)
4. Gleicher, M.: Retargetting motion to new characters. In: Proc. of ACM SIGGRAPH, pp. 33–42 (July 1998)
5. Monzani, J., Baerlocher, P., Boulic, R., Thalmann, D.: Using an intermediate skeleton and inverse kinematics for motion retargeting. In: Eurographics 2000, vol. 19(3) (2000)
6. Shin, H., Lee, J., Shin, S., Gleicher, M.: Computer puppetry: An importance-based approach. ACM Trans. Graph. 20(2), 67–94 (2001)
7. Yamane, K., Nakamura, Y.: Natural motion animation through constraining and deconstraining at will. IEEE Trans. on Visualization and Computer Graphics 9(3), 352–360 (2003)
8. Baerlocher, P., Boulic, R.: An inverse kinematic architecture enforcing on arbitrary number of strict priority levels. The Visual Computer 20(6), 402–417 (2004)
9. Boulic, R., Mas, R., Thalmann, D.: A robust approach for the center of mass position control with inverse kinetics. Journal of Computers and Graphics 20(5) (1996)
10. Kovar, L., Gleicher, M., Schreiner, J.: Footskate cleanup for motion capture. In: ACM Siggraph Symposium on Computer Animation 2002 (2002)
11. Witkin, A., Kass, M.: Spacetime constraints. In: Proceedings of ACM SIGGRAPH, pp. 159–168. Addison Wesley, Atlanta (1988)
12. Liu, C.K., Popović, Z.: Synthesis of complex dynamic character motion from simple animations. ACM Transaction on Graphics 21(3), 408–416 (2002)

13. Abe, Y., Liu, C., Popovic, Z.: Momentum-based parameterization of dynamic character motion. In: Proceedings of ACM SIGGRPAH/Eurographics Symposium on Computer Animation, Grenoble, France, pp. 173–182 (2004)
14. Sulejmanpasic, A., Popovic, J.: Adaptation of performed ballistic motion. ACM Trans. on Graphics 24(1), 165–179 (2005)
15. Chai, J., Hodgins, J.: Constraint-based motion optimization using a statistical dynamic model. ACM Transactions on Graphics - Siggraph 2007 26(3), 8 (2007)
16. Ko, H., Badler, N.I.: Animating human locomotion in real-time using inverse dynamics. IEEE Computer Graphics & Applications (1996)
17. Shin, H., Kovar, L., Gleicher, M.: Physical touch-up of human motions. In: Proceedings of Pacific Graphics, Alberta, Canada (2003)
18. Majkowska, A., Faloutsos, P.: Flipping with physics: Motion editing for acrobatics. In: Proceedings of Eurographics/ACM SIGGRAPH Symposium on Computer Animation, San Diego, USA (in Press, 2007)
19. Multon, F., Hoyet, L., Komura, T., Kulpa, R.: Dynamic motion adaptation for 3d acrobatic humanoids. In: Proceedings of IEEE Humanoids 2007 (2007)
20. Yamane, K., Nakamura, Y.: Dynamic filters - concept and implementations of online motion generator for human figures. IEEE transactions on robotics and automation 19(3), 421–432 (2003)
21. Tak, S., Ko, H.: A physically-based motion retargeting filter. ACM Transactions on Grpahics 24(1), 98–117 (2005)
22. Arikan, O., Forsyth, D., O'Brien, J.F.: Pushing people around. In: SCA 2005: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 59–66 (2005)
23. Zordan, V., Majkowska, A., Chiu, B., Fast, M.: Dynamic response for motion capture animation. ACM Trans. Graph. 24(3), 697–701 (2005)
24. Multon, F., Kulpa, R., Bideau, B.: Mkm: a global framework for animating humans in virtual reality applications. Presence 17(1), 17–28 (2008)
25. Kulpa, R., Multon, F., Arnaldi, B.: Morphology-independent representation of motions for interactive human-like animation. Computer Graphics Forum, Eurographics 2005 special issue 24(3), 343–352 (2005)
26. Tolani, D., Goswami, A., Badler, N.: Real-time inverse kinematics techniques for anthropomorphic limbs. Graphical Models 62, 353–388 (2000)
27. Lander, J.: Making kine more flexible. Game Developer Magazine 1, 15–22 (1998)
28. Kulpa, R., Multon, F.: Fast inverse kinematics and kinetics solver for human-like figures. In: Proceedings of IEEE Humanoids, Tsukuba, Japan, December 2005, pp. 38–43 (2005)

More Motion Capture[⋆] in Games — Can We Make Example-Based Approaches Scale?

Michael Gleicher

University of Wisconsin - Madison, Madison WI 53706, USA
gleicher@cs.wisc.edu
http://cs.wisc.edu/~gleicher

Abstract. Synthesis-by-Example (SBE) approaches have been successful at animating characters in both research and practice (games). These approaches assemble motions from pre-recorded examples, usually motion captured or keyframed. To date, the methods have relied on a small set of simple, generic building blocks for assembling the motions. To meet the increasing demands for better character movement and control in games, the approaches will need to evolve. An obvious path to address these challenges, employing increasingly large collections of examples, is enabled by recent research. However, scaling up the number of examples is unlikely to sufficiently scale up the quality of the character animation. Methods that make better use of examples will be required.

1 Introduction

Human (or human-like) characters are important in many types of computer games and interactive simulations. Often, the movements of these characters are created by an example-based approach where pre-recorded clips of movement, either from motion capture or keyframing, are assembled as needed. While these *Synthesis-By-Example* (SBE) approaches have been extremely successful in research and practice, future applications (e.g. improved games) will have increased demands. In this paper, we consider how the (SBE) approaches may evolve to meet these new needs.

Creating the movements for game characters is difficult. Animating human characters is difficult in general: human movement is incredibly complex, diverse, and subtle. People can do many things in many ways. Even simple, everyday actions like walking involve the complex coordination of many degrees of freedom, and involve an amazing degree of subtlety. Movements can convey a large amount of information in this subtlety: from watching someone walk, we can often get a sense of their mood, their personality, their intent, etc.

Interactivity (e.g. in games) provides another set of challenges. While many games include scripted “cut-scenes,” most game play involves a player’s action (or other unknown factors) and therefore cannot be determined ahead of time. Characters’ movement must be responsive to the unfolding situation in the game,

[⋆] Or keyframe animated motion.

© Springer-Verlag Berlin Heidelberg 2008

whether it is under the direct control of the player or otherwise. Games often involve longer durations requiring large amounts of animation.

There are two main strategies for creating game character motion: algorithmic (model-based) synthesis and synthesis by example (SBE). The key ideas contrast: algorithmic synthesis focuses on creating specialized techniques based on an understanding of movement, while SBE methods employ generic algorithms that avoid understanding the movement, treating it as generic data that is combined by simple, generic procedures. In practice, a spectrum of approaches blends these extremes.

Much of the success of current game character animation stems from the use of SBE approaches. As will be discussed in Section 2, examples provide a number of advantages for animation creation. SBE approaches (§3) have been successful in research and practice as they preserve these advantages. However, future games will need better character animation (§4). Improving the results of an SBE approach typically means using a larger set of examples, but while the enabling technologies for this are in place, this strategy is unlikely to scale (§5). Instead, we are likely to need to improve the methods used in SBE, some first steps in this direction are discussed in Section 6.

2 Why Examples?

The central idea of example-based, or data-driven, approaches to motion is that the movement is obtained from some outside source (like capturing the motion of an actor or having an animator create keyframes). The movement data is just data: a set of measurements that can be replayed. There is no “model” of the movement to explain why a particular set of data is the desired motion. All of the complexity and subtlety is stored in the data.

Examples avoid the need to model the movements to be created. High quality movement can be created without understanding the complexities and subtleties. Given enough time and effort, it may be possible to model a particular motion algorithmically. It is unclear how well the effort for any particular model applies to other motions. Modeling motion is difficult to scale to large repertoires or diversity of styles.

In contrast, example-based approaches readily scale to diverse sets of movement - all that is required is to obtain more examples. An actor (or a keyframe animator) is capable of an amazingly wide range of actions and styles, and can quickly produce many examples of movement. More significantly, example creation provides artistic control. Motion capture involves a partnership and communication between the actor, the director, and (to a lesser degree) the technologist. The actor and director can work together to create the necessary movement, without having to figure out how to explain it in sufficiently concrete terms that it can be encoded algorithmically. The beauty of example-based approaches is that they put creative control over the motion into the hands of the artistic team.

3 Synthesis by Example

By themselves, examples provide only the specific motions that are recorded. Synthesis-by-Example approaches create new motions based on a collection of pre-recorded examples.

Any synthesis approach involves some amount of algorithmic process, and usually data. We use the term *Synthesis-By-Example* (SBE) to refer to approaches that attempt to stay close to the model-free, data-driven concept. The spirit of such approaches is that they tend to be model-free: motions are just sets of numbers that are combined using simple, generic algorithms. The specifics and complexities of different movements come from the data. In contrast, *Model-Based* approaches encode more of the movement algorithmically. SBE approaches maintain the advantages of examples: movements are primarily specified through the examples, allowing for a diverse range of movements to be created, and for these movements to be specified by the artistic team.

Synthesis-By-Example approaches generally combine example motions in a few basic ways. These basic operations, such as blending or sequencing, are not necessarily simple: finding a motion that is "halfway in-between" running and sitting, or a transition between walking and a handstand, can be as difficult as creating the initial examples themselves. However, the space of motions is generally smooth: that is, small changes to valid motions are likely to be valid motions as well. Therefore, if motions are similar, combining them is easy. Basic mathematical operations, such as blending or sequencing, provide reasonable motions when provided with appropriate data.

Synthesis-by-Example methods generally use simple techniques for combining motions, but apply them only where they are likely to lead acceptable results. Almost all SBE is based on the same building blocks: motions are blended (generally by linear combinations of individual parameters), concatenated (often with transitions in between), layered (per-channel adjustments are phased-in and -out), and transformed (positioned spatially and temporally).

Much of the limitations of SBE approaches stem from the simplicity of the building blocks: because the basic techniques are only likely to work on appropriate example motions, their applicability is limited. Since the techniques generally only apply to create small changes to motions (e.g. to transition or blend between similar motions, or to make small adjustments to an example), the achievable results must be similar to the source examples. Phrased differently: without a model to provide an understanding of why an example is desirable, a method must be conservative in how it deviates from the example since it cannot know what in the example must be preserved.

While SBE methods all share the same basic building blocks (e.g. blending, concatenation) for combining motions, they differ in how these building blocks are used. These combination methods are only one of three aspects of an SBE method. Additionally there must be *preparation* to determine what data can be combined and *control* to determine how to use the combinations to assemble the necessary motions. There is much more diversity in these other aspects of SBE.

3.1 Synthesis-by-Example in Games

The use of pre-recorded pieces that are assembled to create interactive animations in games certainly predates the use of motion capture. However, the demands of motion capture pushed the techniques. With motion capture data (or 3D animated characters in general), the pieces needed to be assembled more carefully to maintain visual quality (or at least avoid objectionable artifacts). As the early pioneers developed the use of motion capture in games, they also had to develop the basic foundations of the synthesis-by-example approaches to use these results. Unfortunately, there is little record of these early developments.[1]

By the mid-1990s, synthesis by example systems were applying motion capture to characters in games[2]. Transition graphs were planned out in advance, and manually constructed. Tools for supporting graph design were in use at least far back enough to be used for games released in early 1996 [5], although the first published description of a graph construction tool seems to be much later [6]. Even in this early era, blending was used both to create transitions as well as to create more precise control (e.g. blending left and right to create gradated turns), although most early blending was pre-computed because it was considered too costly to be done at run time[5].

Modern game characters have evolved from their early origins. However, they are still (often) built from the same basic SBE building blocks. Modern characters usually have a discrete set of actions, where each action has a continuous parameterization. For example, a character might be able to punch and kick to various locations and walk with various turning curvatures and speeds. Such characters still employ a transition graph to specify which actions can follow a given one, but the graphs describe parametric ranges of movements, rather than specific examples. The motions for the parametric actions might be created by any number of methods, such as blending examples together or making layered adjustments to a single example.

The key to practical application of synthesis by example approaches has been the careful planning and manual labor in preparing the data such that the examples work together[1,7]. Developers carefully choose what movements to acquire (either capture or hand animate), carefully plan these movements so that they can transition or blend as needed, carefully choose the blends and transitions to facilitate the necessary control over the character, and carefully perform the movements to create examples that can be connected.

[1] Published accounts of the early motion capture, such as [1] or [2,3,4], generally focus on the development of the capture hardware, the application of data to character models, or film applications. Most of my knowledge of this “folklore” of motion editing history comes from conversations with the pioneers. In particular, Mark Schafer has graciously provided me with some specifics of the history at Acclaim.

[2] It is difficult to pinpoint the exact origins, since games used motion capture for cut-scenes and promotional animations as well.

3.2 Synthesis-by-Example in Research

Synthesis-by-Example approaches have evolved differently in research than in practice. The same basic building blocks of blending, layering, and concatenation are used. The introduction of these methods to the research community (for example, blending [8,9,10,11] or layering [12,13]), often came after they had already been deployed in practice. While there are examples of efforts to provide improved building blocks, such as better transition generation [14] or alternate blending schemes [15], most work on synthesis-by-example approaches has focused on making use of the same basic building blocks for combining motions as used in practice.

Where research has deviated from practice is in the application of automation and more advanced algorithms for the preparation of the examples and the creation of control strategies. For example, the "motion graph" approaches [16,17,18] (and their successors) distinguish themselves from their predecessors by automatically searching through a repository of motions to find potential transitions to introduce. This opportunistic graph construction avoids the labor of identifying transitions and offers the potential for reducing the planning effort. However, the unstructured nature of the resulting graphs makes the control problem more complex. Research has addressed this challenge with a wide range of search strategies.

Reducing the need for planning and manual labor through automation and clever control strategies is a common theme in SBE research. For example, [19] shows how the alignments required for blending can be determined automatically. This automation not only saves labor, but allows for the creation of blends that would be impractical to create manually. In an extension of the approach, [20] shows how a database of motions can be searched to find appropriate examples to blend and provides an automatic way to build the control strategy that maps parameters to blending weights. Again, this not only significantly reduces the amount of effort required to create parametric actions, but also extends the range of where blending can be applied. The ease with which parametric actions can be built allows for easy experimentation, which often leads to surprising examples.

Another aspect of SBE that has been automated in research is the design of controls (choosing which examples to combine and how). The search strategies demanded by unstructured graphs have lead to entirely new control paradigms. For interactive control, pre-computed search [21,22] and optimal control [23] methods automate control mappings even in situations where the example collection is not carefully planned. Automatic blend parameterizations [20] allow for precise control of blends even when the example set is irregular.

4 The Needs for Better Game Animation

While character animation in games is quite advanced, improvement is still important. In current games, the quality of the movement of characters already lag other aspects, particularly the visual appearance of the environments. Advances in interactive rendering that exploit increasing hardware resources (GPUs) are widening the gap. If nothing else, movement quality must improve so that characters don't look out of place in the well-rendered scenes.

Characters with more convincing movement can better add to creating compelling visuals, and provide game designers with more flexibility to create a wider range of compelling experiences for players. However, to provide for this range of future games, the technology to animate character must provide:

More quality - the characters' movements should meet the designer's goals for the visual style that creates the experience. For example, if a game's design is meant to be realistic, the characters in the game should move realistically.

More actions - characters should have richer repertoires, and ultimately be able to do (at least) the range of things that actors can do.

More styles - characters should be able to do these things in the entire range of ways that people do them, as these differences are often important.

More subtlety - the differences in the ways that things are done can often be quite subtle. Conveying these subtleties is important for communicating things such as mood, intent and personality. Current games generally rely on other means (such as narration) to convey this.

More situated - the characters' movements must relate correctly to their surroundings, otherwise the illusion of the character inhabiting its world is broken. This requires a degree of precision in motions: if connections (such as contacts) are not exact, they are a very visible reminder.

More responsiveness - characters should respond quickly to their control (e.g. a player's commands or other occurances in their environment).

While the current technologies allow for excellence in some of these categories, this excellence usually comes at the expense of other attributes. For example, it is possible to make a character with very high quality movement if its range of actions is very small, or is unresponsive - the cut scenes in games often have very good motion.

5 Using More Examples

The results of SBE can be improved by using larger sets of examples. Larger sets of examples help improve many of the aspects of character animation:

More actions and styles - most obviously, having a wider range of examples is the primary (possibly the only) way to extend the range of actions and styles the character is capable of.

More quality, subtlety and artistic control - more examples means that whatever motion is going to be created is more likely to be close to an example. Deviation from the examples is the source of loss of quality (because the examples are given by the artistic team and express their goals).

To see how increased example sets can help, consider a simple example: a character walks up to a bookcase and grabs a book. With a single example, an SBE approach might use IK to reposition the hand to reach different places on the book shelf, and then to propagate these changes to neighboring frames. If many examples are provided (i.e. having examples of a person reaching to multiple locations), the other variability in the movements can be captured. For example, a person might move differently to read a higher object, and their eyes need to find the book before they can grab it. While it may be possible to algorithmically encode these details, each would need to be identified, understood, and implemented.

More responsiveness - a larger example set provides more options for synthesis methods to create motions, so it is more likely that a choice will be readily available when a change is necessary.

Fortunately, technological improvements have facilitated obtaining and using larger example sets. The equipment and software for motion capture and post-processing makes obtaining more examples practical. The wider availability of the equipment reduces the limitations in the amount of capture possible. At run-time, data storage is becoming more plentiful and motion data is very compact relative to other assets like sounds and textures.

Automation in the authoring tools, as described in Section 3.2 are better able to scale to larger number of examples. Automatic tools not only reduce the amount of labor to process the data, they also reduce the required planning and can provide results that cannot be achieved manually, such as very complex blend spaces, accurate blend parameterizations, or near-optimal control.

5.1 Why Example Sets Cannot Scale

One possible stumbling block for increased motion usage is performance. While modern games (especially consoles) have considerable processing resources, they are often constrained in the amount of bandwidth available to access examples. Also, motion synthesis is only one aspect of a game whose computing resources are growing. Increased computational abilities have raised player expectations about rendering quality, visual complexity, and artificial intelligence, all of which demand increasing amounts of processing resources. To retain practicality, the memory bandwidth requirements of SBE methods needs to be considered more thoroughly as example sizes scale.

A more challenging issue is unintended variation in the examples. Some of this comes from limitations in the combination processes. For example, in the bookcase scenario above, in order for blending to work, the character must always initiate the reaching motion with a step on the same foot. More subtle variabilities may not cause failures, but instead have unintended consequences. For example, imagine a walking character created by combining footsteps found in an example database. While the large example set may provide a diverse range of steering and speeds allowing for a very controllable character, every footstep has its own story. For example, on any given example the actor might be more or less tired or distracted, have a more or less clear idea of where they are going, or may have stumbled or twitched. As SBE chooses different examples for each step, it may mix these stories[3]. The degree of quality assurance to insure the regularity of the examples may preclude large example sets.

All of the variabilities could be viewed positively: all of the differences between motions might become parametrically controllable. There are several reasons why such an approach is unlikely to scale. First, existing methods for automation are not good at identifying and parameterizing the more subtle variability. Second, the different parameters aren't necessarily orthogonal. Third, tradeoffs between differences are difficult to compare (is it better to pick an example that is similar in tiredness, personality, or position?). Fourth, as the number of parameters grows, the space of possibilities grows exponentially. This leads to increased demands on the number of examples required to adequately sample the space and the methods for controlling within it.

The inability of SBE approaches to deal with high dimensional parameter spaces is likely to be the ultimate limitation on its scalability. To create truly expressive and responsive characters, a myriad of properties need to be controlled for any particular action. In games, it is easy to see a slippery slope of wanting more and more parameters to be controlled: a walking character should be able to turn, vary its speed, step up/down on obstacles, have varying levels of injury, have varying levels of intensity/focus, ...

The curse of dimensionality is a final limiting factor of the standard SBE approaches. While automation might help the methods scale to larger example sets, it cannot help them scale enough. Example sets would need to grow exponentially.

6 Scaling SBE Methods

The previous section argued that increasing the size of the example sets used in SBE methods is unlikely to scale to the needs of future applications. Similar arguments have been made by several others in this workshop (c.f. [24,25,26]).

The power of example-based methods to allow for artistic collaboration to specify desired movement properties by example means that the SBE approach is unlikely to go away. Purely algorithmic approaches, in some sense the antithesis of example-based ones, still must somehow engage collaboration with the artists, designers, and directors who provide the vision of the movement requirements. While Perlin has shown great progress in creating parameterized algorithmic controllers at this workshop [25], it is unclear how well this approach will scale to large repetoires, movement styles, or ranges of visual style (including realism). It takes a very expert programmer to understand movement well enough to devise algorithmic synthesis processes, and the need to make these flexible enough to meet an artists' stylistic wishes even further complicates the problem.

I believe that the future of technology for animated characters in interactive systems lies as a hybrid of synthesis-by-example and algorithmic approaches.

[3] Sometimes, coherence in the variability can lead to unusual outcomes. In one capture shoot, left turns were captured in the morning and right turns at the end of the day, yielding a character that looked tired whenever they turned right.

There are two different ways in which the “pure” approaches may mix: using more sophisticated methods for combining examples and using collections of data to derive algorithmic controllers. Examples of both paths can be seen in the literature. Popović’s presentation at this workshop [26] provides a particularly compelling example of how an algorithmic controller might be derived from examples.

Here, I provide two brief examples from our group’s work[4] that illustrate these hybrid approaches. In the first example, the standard set of methods for combining examples is extended, providing a mechanism for scaling to a broader repertoire without a commensurate expansion in the example set. In the second example, an algorithmic synthesis procedure is derived from data. While the strategies are quite different, they both represent attempts to provide more scalable SBE approaches.

6.1 Splicing Actions

Different parts of a character might perform different actions simultaneously. For example, a character might wave, carry a box, or stare in a particular direction at the same time that they walk, stand or sit. This creates a potential combinatorial explosion of possible things a character might do (i.e. stare to the left while standing, carrying a box, tapping the left foot). When limited to the traditional mechanisms for combining examples, examples are needed for each combination.

Being able to partition the parts of a character and provide independent examples (or motion synthesis methods) for each avoids the need for all combinations. For example, if we could consider the upper and lower body separately, we could have a set of example upper body actions (e.g. wave, salute, carry a box, hold a coffee cup) and lower body actions (e.g. walk and run with various turns and speeds) without having examples of all n^2 combinations. *Splicing* methods assemble movements for a character from independent sources of movement for each part.

Adding splicing to the set of methods used to create SBE offers a mechanism for greatly reducing the number of examples needed. However, when multiple actions are performed simultaneously, they do interact (e.g. carrying a heavy box changes the way one walks). Creating these couplings can be challenging. Simple splicing methods do not provide the proper couplings, and therefore often look wrong. However, splicing is so useful that these simple splicing methods are commonly used in practice in games despite the quality problems.

The diversity and complexity of couplings between body parts suggests that a general solution for splicing may be illusive. To date, researchers have focused on creating splicing methods for specific parts and situations, such as hands [28]. In [29] we presented a method for splicing upper body motions onto lower bodies in the specific case where both examples come from locomotion. The method works by specifically identifying important types of couplings, including posture, coordinated timing, and spatial alignment, and taking specific steps to make sure that each coupling is properly established in the result.

[4] The research described in these sections is part of the Ph. D. thesis research [27] of my former student, Rachel Heck.

Splicing can serve as a building block for synthesis-by-example, along side the more usual blending and concatenation. By building SBE approaches with a richer set of building blocks, there is a potential to achieve greater performance (in terms of result quality, repetoire range, range of stles, etc.) without an explosion of examples. However, our splicing method lacks the simplicity, genericness, and broad applicability of the "pure" SBE approaches. In a sense, our splicing technique shows a mixture of a model-based and example-based approach: understanding of a specific class of motions was used to create an algorithmic synthesis method that relies on data.

6.2 Gaze Control

It is important to be able to control the gaze direction of an animated character. To an observer, shifts in the location or direction where a character is looking might not only indicate a shift in attention but can also convey a person's goal before they act on it.

Effective control of the gaze direction is complicated. The gaze direction is determined by the orientation of head, as well as the eyes. To look in a particular direction, a person might adjust their torso and neck (in order to orient the head), as well as move their eyes. The timing and coordination of these movement are also complicated, as the eyes can move much more rapidly than the head, leading to a progression where they move first, and usually overshoot the target. The specifics of the movements depend on the direction, the size of the change, and even the individual and their mood (e.g. different people have different ranges of motions and preferences, and may react differently if they are tired or scared).

Gaze control would be very difficult with standard SBE approaches. It adds at least 2 new parameters to any movement (the direction of gaze). Providing good control would require not only sufficient examples to allow for the range of gaze directions, but also to allow for the range of gaze timings (i.e. a character might look in a particular direction at a particular instant). The number of examples required to create such a diverse space of possibilities would be prohibitive.

We have developed a technique for controlling the gaze of an animated character, described in Chapter 5 of [27]. Given the character's motion and a gaze target (a direction at a particular time), the motion is adapted to meet the gaze target. The technique uses a specifically designed model of gaze motions, built from an understanding of the psychological and physiological principles involved. In many ways, the technique shows the traditional process of algorithmic synthesis development: where a programmer gained an understanding of a particular movement and encoded this understanding into an algorithmic process with appropriate controllability. In this case, the algorithmic synthesis produces a change to an existing motion (that is added by layering).

Our gaze technique also employs an example-based approach to achieve individual and/or mood/style variability. Motion capture data of an actor performing a number of examples of gaze movement is used to generate a set of parameters that are used by the gaze controller. Effectively, the algorithmic model is built from example data. The gaze technique exemplifies the broader goal of deriving algorithmic control is from example data.

7 The Future

A great actor provides a director/producer with an amazing range of possible actions and movement styles. Computer animated characters extend this flexibility in other ways, such as providing for different visual styles (e.g. realistic, cartoony) or responsiveness tradeoffs. The greater range of character animation makes more tools available for designers to create better interactive experiences. The technology to drive future game characters will need to be more like a great actor, providing game developers with a powerful and expressive component to create better games.

These future game characters will require technology beyond what is currently available. Examples are still likely to be useful, as they enable a designer or director to specify the movements that they want, as well as providing an effective way to create the necessary diversity of actions and styles. However, current synthesis-by-example approaches with larger sets of examples are unlikely to scale to meet to meet the challenges. New methods that can make more use out of a compact set of examples will be required.

Acknowledgements

The research work performed in our group represents the efforts of a long list of talented students, including Lucas Kovar, Rachel Heck (now Rose), Alex Mohr, Andrew Selle, Mankyu Sung, and Hyun Joon Shin.

References

1. Menache, A.: Understanding Motion Capture for Computer Animation and Video Games. Morgan Kaufmann, San Francisco (1999)
2. Elson, M., Sturman, D., Dyer, S., Trager, W., Schafer, M.: Character motion systems (SIGGRAPH, Course Notes) (1994)
3. Sturman, D.: A brief history of motion capture for computer character animation. SIGGRAPH Hypergraph Web Page (1994)
4. Trager, W.: A practical approach to motion capture: Acclaim's optical motion capture system. SIGGRAPH HyperGraph Web Page (1994)
5. Schafer, M.: Personal Communication (July 2008)
6. Mizuguchi, M., Buchanan, J., Calvert, T.: Data driven motion transitions for interactive games. In: Eurographics 2001 Short Presentations (September 2001)
7. Kines, M.: Planning and directing motion capture for games. Game Developer Magazine (1998); Also in GDC 1998 and Gamasutra
8. Perlin, K.: Real time responsive animation with personality. IEEE Transactions on Visualization and Computer Graphics 1(1), 5–15 (1995)

9. Guo, S., Roberge, J.: A high-level control mechanism for human locomotion based on parametric frame space interpolation. In: Proc. of Eurographics Workshop on Computer Animation and Simulation 1996, pp. 95–107 (August 1996)
10. Wiley, D., Hahn, J.: Interpolation synthesis of articulated figure motion. IEEE Computer Graphics and Application 17(6), 39–45 (1997)
11. Rose, C., Cohen, M., Bodenheimer, B.: Verbs and adverbs: multidimensional motion interpolation. IEEE Computer Graphics and Application 18(5), 32–40 (1998)
12. Witkin, A.P., Popović, Z.: Motion Warping. In: Proc. SIGGRAPH 1995, pp. 105–108 (August 1995)
13. Bruderlin, A., Williams, L.: Motion signal processing. In: Proc. ACM SIGGRAPH 1995. Annual Conference Series, pp. 97–104 (1995)
14. Rose, C., Guenter, B., Bodenheimer, B., Cohen, M.F.: Efficient generation of motion transitions using spacetime constraints. In: Proc. SIGGRAPH 1996, pp. 147–154 (1996)
15. Mukai, T., Kuriyama, S.: Geostatistical motion interpolation. ACM Transactions on Graphics 24(3), 1062–1070 (2005)
16. Arikan, O., Forsyth, D.A.: Synthesizing Constrained Motions from Examples. ACM Transactions on Graphics 21(3), 483–490 (2002)
17. Kovar, L., Gleicher, M., Pighin, F.: Motion graphs. ACM Transactions on Graphics 21(3), 473–482 (2002)
18. Lee, J., Chai, J., Reitsma, P., Hodgins, J., Pollard, N.: Interactive control of avatars animated with human motion data. ACM Trans. on Graph. 21(3), 491–500 (2002)
19. Kovar, L., Gleicher, M.: Flexible automatic motion blending with registration curves. In: Proceedings of the Symposium on Computer Animation (July 2003)
20. Kovar, L., Gleicher, M.: Automated extraction and parameterization of motions in large data sets. ACM Transactions on Graphics 23(3), 559–568 (2004)
21. Lee, J., Lee, K.H.: Precomputing avatar behavior from human motion data. In: SCA 2004: Proc. of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 79–87 (2004)
22. Lau, M., Kuffner, J.J.: Precomputed search trees: Planning for interactive goal-driven animation. In: 2006 ACM SIGGRAPH / Eurographics Symposium on Computer Animation, pp. 299–308 (September 2006)
23. Treuille, A., Lee, Y., Popović, Z.: Near-optimal character animation with continuous control. ACM Trans. Graph. 26(3), 7 (2007)
24. Thalmann, D.: Motion modeling: Can we get rid of motion capture? In: Egges, A., Kamphuis, A., Overmars, M. (eds.) MIG 2008. LNCS, vol. 5277, pp. 121–131. Springer, Heidelberg (2008)
25. Perlin, K., Seidman, G.: Autonomous digital actors. In: Egges, A., Kamphuis, A., Overmars, M. (eds.) MIG 2008. LNCS, vol. 5277, pp. 246–255. Springer, Heidelberg (2008)
26. Popović, Z.: Towards robust dynamic controllers for high-fidelity character locomotion. In: Egges, A., Kamphuis, A., Overmars, M. (eds.) MIG (2008)
27. Heck, R.: Automated Authoring of Quality Human Motion for Interactive Environments. PhD thesis, Dept. of Comp Sci., University of Wisconsin (2007)
28. Majkowska, A., Zordan, V., Faloutsos, P.: Automatic splicing for hand and body animations. In: Proc. of the Symposium on Computer Animation (SCA) (2006)
29. Heck, R., Kovar, L., Gleicher, M.: Splicing upper-body actions with locomotion. Computer Graphics Forum 25(3) (2006); Proc. Eurographics

Simulating Interactions of Characters

Taku Komura, Hubert P.H. Shum, and Edmond S.L. Ho

Institute of Perception, Action and Behaviour
School of Informatics, University of Edinburgh

Abstract. It is difficult to create scenes where multiple characters densely interact with each other. Manually creating the motions of characters is time consuming due to the correlation of the movements between the characters. Capturing the motions of multiple characters is also difficult as it requires a huge amount of post-processing of the data. In this paper, we explain the methods we have proposed to simulate close interactions of characters based on singly captured motions. We propose methods to (1) control characters intelligently to cooperatively / competitively interact with the other characters, and (2) generate movements that include close interactions such as tangling the segments with the others by taking into account the topological relationship of the characters.

Keywords: character animation, motion capture, crowd simulation.

1 Introduction

Scenes that multiple characters densely interact with each other are common in daily life. Such scenes include multiple people carrying luggage together, one person holding the shoulder of another injured person and helping him/her walk, a group of people fighting or playing sports such as wrestling, rugby, or ice hockey, and people dancing in a densely crowded hall sticking their body together. Controlling each character under such environment is difficult as the motion of one character affects the motions of all the people in the scene.

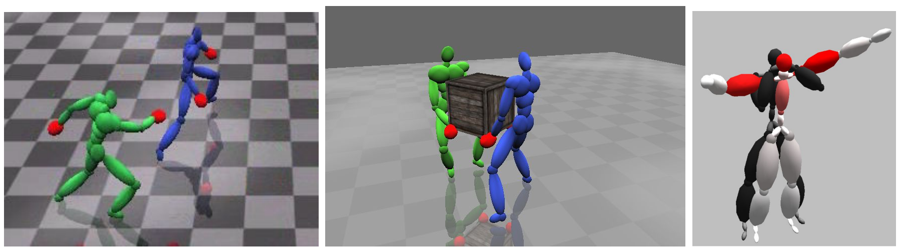


Fig. 1. The close interactions of human characters that we cover in this paper: (a) A character chasing another character taking into account the benefits in the future, (b) two characters carrying luggage together, and (c) two characters wrestling

© Springer-Verlag Berlin Heidelberg 2008

Currently, the cost and time required to create such scenes are enormous. Such motions need to be either created manually or using the motion capture system.

Manually creating a scene of multiple characters is time consuming as each movement of the characters is correlated with those of the others. One amendment results in a number of updates in the scene. Suppose an animator is editing a scene where one character is knocking down another character. If we need to edit the punching motion of the attacker, by changing the position and timing of the punch landing onto the opponent, then we also need to edit the motion of the opponent being knocked down, by changing the way it gets hit and falls down onto the ground. If the two characters are repeatedly attacking / defensing, the amount of work becomes enormous.

Capturing the motions of multiple persons in the scene at the same time using a motion capture system is another solution to create such an animation; however, for scenes as fighting, this is difficult due to the intrusiveness of the motion capture system, occlusions, and the intensive interactions that affect the performance of capturing. Think of using an optical motion capture system to capture two boxers seriously sparring. In the beginning, they actually never seriously hit each other as markers are attached to various parts of their bodies. Athletes are sensitive to such an unusual condition and they cannot perform in the way they usually do. Although it is difficult, suppose the boxers overcome such pressure and start to spar at close distance seriously. A lot of markers are occluded by the arms, head and torso of each boxer as they are hitting each other in a very close distance. And finally it will be found out capturing serious intense sparring is almost impossible as the markers are flying away from their bodies when one fighter's arm hits/rubs the surface of the other's body. The situation will be similar or even worse when magnetic / mechanical motion capture systems are used, as they are even more intrusive than the optical system.

Therefore, we take a more practical approach; we capture the motion data individually, and simulate the interactions by controlling the characters using AI techniques. In that case, the problems of occlusions and post-processing become much easier to handle. We can also produce combinations of actions which are difficult to be captured due to safety. In this paper, we introduce our previous work to generate animation of multiple characters closely interacting with each other. We simulate competitive interactions such as chasing, fighting and playing sports, as well as cooperative interactions such as carrying luggage together. Three methodologies are presented:

1. **Simulating Competitive Interactions using Singly Captured Motions (Section 2).** First we explain a method to generate a realistic scene of characters interacting in a competitive environment with a method similar to controlling computer-based players in chess. We expand the game tree and evaluate the interactions of characters taking into account the rewards in the future. This method is effective to simulate scenes such as fighting and chasing.
2. **Simulating Interactions of Avatars in High Dimensional State Space (Section 3).** Although the above approach is effective to simulate

realistic interactions between the characters, it is difficult to apply it to real-time applications such as 3D computer games. This is because game-tree expansion requires exponential computational time. Here we explain a method to create and utilize a finite state machine which is effective to control computer-controlled characters in real-time.

3. **Creating tangled motions (Section 4).** In the previous approaches, we mainly focused on instantaneous and impulsive interactions. In some cases, we need to handle interactions where a character tangles its body segments to those of the others, such as when piggy-backing another person, giving shoulder to an injured person to walk, or playing wrestling. Here we explain a method to create such motions by taking into account the topological relationships of the characters.

2 Simulating Competitive Interactions Using Singly Captured Motions

In this section, we explain the overview of our method to simulate interactions of characters by expanding the game-tree and evaluating the status of the characters in the future. For the details of the techniques, the reader is referred to [8]. By using this method, the characters make intelligent choices taking into account their rewards in the future. As a result, the interactions between the characters appear more realistic than when using emergent approaches such as flocking.

The outline of the method is shown in Figure 2. We first capture the motions of subjects individually using optical motion capture systems (Figure 2, left-most). The captured motions are segmented and classified into semantic groups such as "straight punch", "kick" or "parry" automatically. Using the annotated motions, we generate a data-structure called action-level motion graph (Figure 2, left middle), which is a motion graph structure whose actions are annotated. In order to control the characters in an intelligent way, it is necessary to make the character predict how the opponent will react to its action, and decide the next action based on its benefits in the future. In order to do this, we expand the game tree, and assess the status in the future using an evaluation function (Figure 2, right middle). Finally, the character selects an action that maximizes its rewards in the future, and launches its action (Figure 2, right most).

We propose a new algorithm called the temporal expansion approach which maps the continuous action planning to a discrete space so that turn-based evaluation methods such as min-max algorithms and $\alpha - \beta$ pruning can be used.

In order to simulate realistic interactions of the characters, we propose to use a table that pairs actions. In this table, the appropriate actions that need to be launched when the opponent character is undergoing some specific actions are listed. For example, for each entry of the attack, the appropriate defense motions together with the best timing to launch them are listed.

The users can easily specify how the scene should appear by tuning parameters. Every character is guided by an objective function, and it is possible to

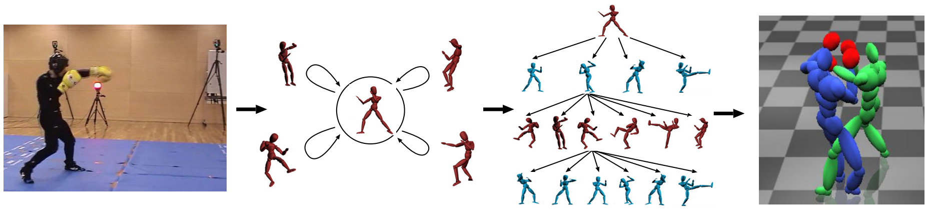
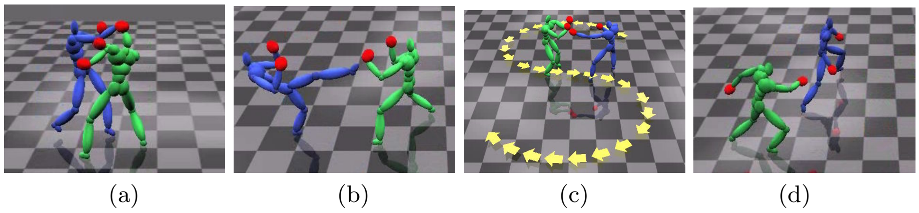


Fig. 2. The outline of the proposed method to simulate competitive interactions: (1) capture the motions of characters individually (2) generate the action level motion graph (3) evaluate the interaction by expanding the game tree (4) simulate the competition by physically-based animation

set up a scenario of the competition, or control the way the character competes by tuning the parameters of its objective function. For example, in case of fighting, it is possible to simulate various fighting styles, such as being more passive, aggressive, or preferring kicks than punches by changing the scores given to the characters when they successfully attack or defend. By giving higher scores to both characters when they follow a path while fighting, both the characters will tend to do so.

2.1 Experimental Results

We have simulated various competitive interactions of the characters to show the effectiveness of our method. We have created examples of boxing matches. The strength of each fighter can be adjusted by changing the depth of the game tree expanded. We can also adjust the parameters of the characters to simulate different styles of fights, including outboxing and infighting (Figure 3 (a),(b)).



(a) (b) (c) (d)

Fig. 3. Some of the screen shots of the simulated fights: (a) Infighters fighting at very close distance, (b) outboxers at long-range distance, (c) the fighters following a path while fighting, and (d) one avatar chasing another

Next, a scene two fighters moving along a predefined path while fighting was simulated (Figure 3 (c)). The path is modeled as a series of check points.

Finally, a scene where an avatar chases another was simulated (Figure 3 (d)). The movements of both avatars are based on the running-around motion. The preferred distance of the chaser is set short and that of the avatar who is running away long. Moreover, based on the scoring function, high score is given to the

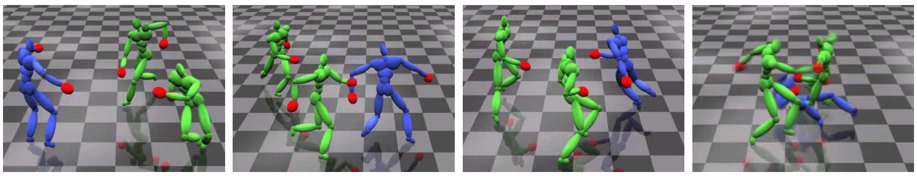


Fig. 4. Two green avatars chasing the blue avatar. The green avatars cooperate with each other to catch the blue avatar.

chaser when it catches the other avatar. As a result, the chaser tries to approach its opponent while the opponent tries to get away. We also simulated a scene where two avatars chase one avatar. In this case, the game tree is composed of nodes and edges which represent the actions of three avatars. The chasers cooperate with each other to catch the avatar who is running away (Figure 4).

3 Simulating Interactions of Avatars in High Dimensional State Space

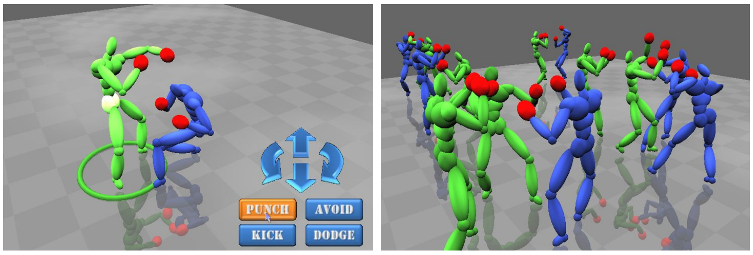
In this section, we explain our method to control non-player-characters (NPCs) of 3D computer games to intelligently interact with human-controlled characters in real-time. For the details of the techniques, the reader is referred to [7].

The method based on game-tree expansion explained in the previous section requires exponential computational cost. As a result, it is difficult that method for controlling NPCs in 3D computer games.

Reinforcement learning enables real-time optimal control of characters. It has been used to control pedestrians to avoid other obstacles/characters walking in the streets [3,10], control a boxer to approach and hit the target [4], make the transition of actions by the user-controlled character smooth [5] and train a computer-controlled fighter in computer games [1]. However, there are two problems that we face when we try to use reinforcement learning to control human characters intelligently when they are interacting with another character.

First of all, the state space is too large. The state space increases exponentially proportional to the number of parameters. Parameters such as the action the character is undertaking, its body position and orientation, and the timing to launch the action are going to form the state space. The number of parameters is going to double if there are two characters. As a result, it is difficult for existing adaptive learning techniques such as Q-learning [11] to explore the whole state space to search for optimal policies.

Another problem is that the way the people behave change according to various factors such as their mood, habits, and preferences of actions; however, previous animation techniques used "on-policy" [9] reinforcement learning methods, which require the system to be retrained in case the reward function is changed. For example, in boxing, there are boxers called infighters who prefer to fight aggressively in short distance, and use punches such as upper cuts and hooks more. On the contrary, there are outboxers, who prefer to stay away from the



(a) (b)

Fig. 5. The interactions of articulated characters are generated by maximizing the reward function defined by the relative pose between characters, the effectiveness of actions, and/or user-defined constraints. This framework of synthesizing character animation is efficient and flexible enough to make a variety of practical applications including (a) interactive character control using high-level motion descriptions such as punches, kicks, avoids and dodges and (b) real-time massive character interactions by a large number of automated characters

opponent and as a result, prefer to use straight punches which are effective in long distance. If we train a virtual boxer by an on-policy reinforcement learning approach, it will not be able to compete well with other fighters who have different styles of fighting. The system needs to be pre-trained for various types of fighters, and the policy needs to be switched according to the type of the opponent, which will be very computationally costly.

Here we make use of the fact that the subspace of meaningful interactions is much smaller than the whole state space of two characters. We efficiently collect samples by exploring the subspace where dense interactions of the characters exist and favoring samples which have high connectivity with other samples. Using the samples collected, a finite state machine (FSM) called Interaction Graph is composed. The Interaction Graph is a Motion Graph of two characters. In order to control the character in an optimal way, a min-max search / dynamic programming is conducted on the Interaction Graph.

We can simulate various activities by two characters such as fighting, chasing, playing sports, or carrying luggage together. Our method can plan strategic movements for Non-Player Characters (NPCs) in 3D computer games. For example, we can control virtual fighters in boxing games (Figure 5(a)), or the background crowd moving or fighting with each other in computer animations (Figure 5(b)), or characters collaboratively working, such as carrying a box (Figure 1(b)).

3.1 Outline of the Method

The procedure of our method can be divided into the preprocessing stage and run-time stage. The **preprocessing stage** proceeds as follows:

1. Capture the motions of a single person conducting the target motion and generate the action level motion graph structure out of the motion data

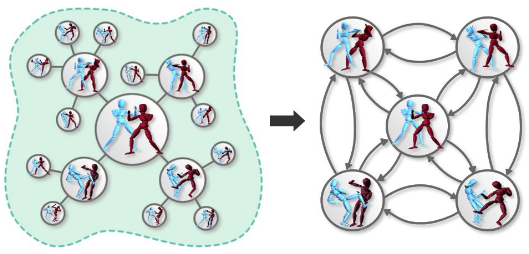


Fig. 6. The outline of the preprocessing stage: (left) exploring the state space by favoring states with more interactions and transitions to existing samples (right) an Interaction Graph created from the collected samples

2. Explore the combined state space of two characters by simulating the interactions of the two characters and expanding the motion tree (Figure 6 left)
3. Generate the Interaction Graph of the two characters and find the most appropriate movements of the characters at each node by dynamic programming or min-max search. (Figure 6 right)

Then during **run-time**:

1. At each state, the corresponding character selects the precomputed optimal action
2. If the animator/user wants to change the policy/strategy of the control, the information in the lookup-table is recomputed by re-running dynamic programming or min-max search. This can be done in a few seconds, and can be run in background while simulating the interactions

3.2 Experimental Results

We have simulated scenes of fighting as examples of competitive interactions and scenes of carrying luggage together as examples of collaborative interactions.

In order to show the real-time performance of our system, we have implemented a game-style interface which the user can control an avatar to fight with the computer-controlled avatar (Figure 7 (a)).

For the example of carrying luggage, another interface to move the avatars to arbitrary directions to avoid being hit by the ball was implemented. Screen shots of this example are shown in Figure 7 (b),(c).

4 Creating Tangled Motions

In this section, we explain our method to simulate close interactions that requires human characters to tangle its limbs with those of the others. We propose a method to use Gauss Integrals (GI), which is a concept proposed in knot theory. For the details of the techniques, the reader is referred to [2].

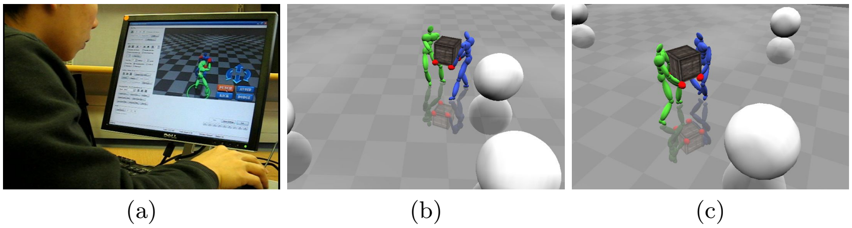


Fig. 7. (a) Using the game style interface, the user can control an avatar to fight with the computer-controlled avatar by the Interaction Graph. (b),(c) Screen shots of the avatars controlled to avoid the ball while holding a box.

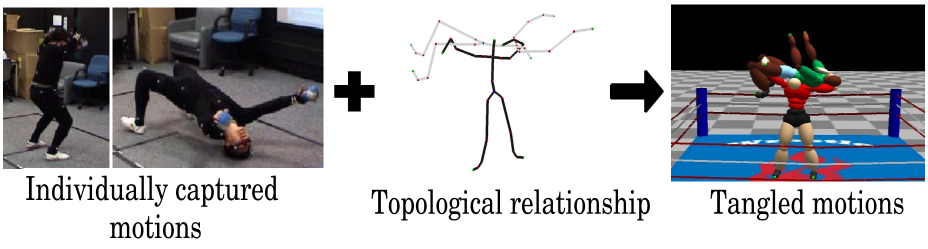


Fig. 8. The outline of the proposed method to simulate interactions of characters that involve tangling

Animations of two characters tangled with each other often appear in battle or fighting scenes in films or games. However, creating such scenes is difficult due to the limitations of the tracking devices and the complex interactions of the characters during such motions. Here we propose a new method to generate animations of two persons tangled with each other based on individually captured motions. We use wrestling as an example.

We propose to use GI to calculate the tangled status of bodies. Since GI can only calculate the relationship of two strands, we propose a method to apply it to express the tangled status of multibody structures such as humans. Once the relationship is specified, the motions of the characters are imported and edited automatically, so that constraints due to penetration or geometrical constraints such as keeping the support feet onto the ground are satisfied. The tangle relationships are also monitored so that the segments do not get untangled. As a result, a scene of two characters interacting with each other can be generated.

Our method can be used to create motions such as holds and chokes in wrestling, a helper holding a shoulder of an injured person to walk or a person piggy-backing another person. The motions created using this method can be used for applications such as computer games and 3D computer animation.

4.1 Methodology

The overview of our methodology is as follows:

1. The user captures the motions of the two characters individually using a motion capture system
2. The user specifies the topological relationship of the characters by composing a template posture with our 3D character posing interface. The postures are examined by the system and the segments composing the tangles are detected by calculating the GLI of the segments.
3. The motion data of both characters are edited according to the topological relationship specified in Step 2.

The flowchart of the algorithm is shown in Figure 8.

4.2 Experimental Results

We have simulated a number of wrestling motions including the Argentine Back-Breaker (Figure 9 (a)), the Rear-Chokehold (Figure 9 (b)), and the Octopus Hold (Figure 9 (c)).

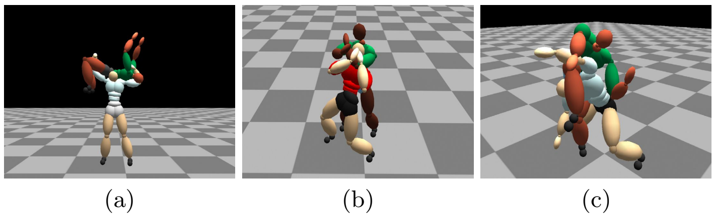


Fig. 9. (a) Argentine Back Breaker, (b) The Rear-Chokehold and (c) the Octopus Hold simulated using our method

5 Summary and Future Work

In this paper, we introduced methods that we have previously proposed to simulate the close interactions of multiple characters. We have covered interactions such as chasing, fighting, carrying luggage, and wrestling. We believe the following topics are the important areas to further explore:

Simulating scenes where a large number of characters interact, such as one person fighting with many background characters, characters fall onto others like domino in panic, and multiple characters pass luggage to the person standing next to it.

Creating a motion graph based on topology: In motion graphs, usually the Euclidean distances of the state vectors based on joint angles or the joint positions are used to evaluate the similarity of different postures. Such kind of distance measures can cause troubles when we animate motions such as wrestling. We can make use of the topological relationship of the bodies in evaluating the similarity of postures of two characters, and compose a motion graph based on this distance measure. It is expected that less penetration of the segments will occur.

Parameterizing the interactions: Currently, we select the actions from a large set of motions. This increases the state space as there can be a set of similar motions which are have the same effect to the scene. We can parameterize the actions [6] from a small set of actions by using interpolation and produce various interactions out of them.

References

1. Graepel, T., Herbrich, R., Gold, J.: Learning to fight. In: Proceedings of Computer Games: Artificial Intelligence Design and Education (CGAIDE 2004), pp. 193–200 (2004)
2. Ho, E.S.L., Komur, T.: Wrestle alone: Creating tangled motions of multiple avatars from individually captured motions. In: Proceedings of Pacific Graphics 2007, pp. 427–430 (2007)
3. Ikemoto, L., Arikan, O., Forsyth, D.: Learning to move autonomously in a hostile world. Technical Report No. UCB/CSD-5-1395, University of California, Berkeley (2005)
4. Lee, J., Lee, K.H.: Precomputing avatar behavior from human motion data. In: Proceedings of 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 79–87 (2004)
5. McCann, J., Pollard, N.S.: Responsive characters from motion fragments. ACM Transactions on Graphics (SIGGRAPH 2007) 26(3) (August 2007)
6. Mukai, T., Kuriyama, S.: Geostatistical motion interpolation. ACM Trans. Graph. 24(3), 1062–1070 (2005)
7. Shum, H.P.H., Komura, T., Yamazaki, S.: Simulating interactions of avatars in high dimensional state space. In: ACM SIGGRAPH Symposium on Interactive 3D Graphics (i3D) 2008 (2008)
8. Shum, H.P.H., Komura, T., Yamazaki, S.: Simulating competitive interactions using singly captured motions. In: Proceeedings of ACM Virtual Reality Software Technology 2007 (2007)
9. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
10. Treuille, A., Lee, Y., Popovic, Z.: Near-optimal character animation with continuous control. ACM Transactions on Graphics 26(3) (2007)
11. Watkins, C.: Learning from Delayed Rewards. PhD thesis, Cambridge University (1989)

Motion Prediction for Online Gaming

Rynson W.H. Lau[1,2] and Addison Chan[1]

[1] Department of Computer Science, University of Durham, United Kingdom
[2] Department of Computer Science, City University of Hong Kong, Hong Kong
rynson@cs.cityu.edu.hk, a.s.k.chan@durham.ac.uk

Abstract. In a multiplayer online game, multiple players located at different geographical locations may participate and interact with each other within a shared game scene through the Internet. Unfortunately, online games generally suffer from the network latency problem, in particular when the players need to interact with each other. We have been developing a geometry streaming environment to support online gaming. As a player moves around in the game scene, geometry information relevant to the player are dynamically sent to the client machine. Such a streaming environment has many advantages, but it also exacerbates the network latency problem in online gaming. In this note, we look at how motion prediction may be used to support multiplayer online gaming and discuss some problems and issues needed to be addressed. We first present existing work on motion prediction. We then discuss our previous work on short-term prediction and our current work on long-term prediction to support online gaming. Finally, we also discuss our work in combining short-term prediction with long-term prediction for geometry prefetching.

Keywords: Motion prediction, online gaming, geometry streaming.

1 Introduction

Multiplayer online games have become very popular in recent years, as they allow users of different geographical locations to participate and collaborate in a game mission through the Internet. However, due to the existence of relatively high network latency of the Internet, the action or movement of a player in the game scene of a multiplayer online game may not be shown immediately to other relevant players in the game. For example, in Final Fantasy XI [1], which is one of the popular online games currently available in the market, it may take a second for a player to receive position updates of other game players. To make the situation worst, each pair of players may suffer from a different amount of latency and this latency value also fluctuates in time. Hence, different players may perceive a change at different times. In order to reduce the effect of such delay in Final Fantasy XI, some restrictions are imposed on the game itself. First, players can only attack enemy objects, but not each other. Second, the enemy objects are designed to move very little while they are under attack by a player. These game rules, however, significantly limit the game features and the type of games that can be developed.

There are two fundamental techniques to address the network latency problem, motion prediction and motion synchronization. Motion prediction methods attempt to

© Springer-Verlag Berlin Heidelberg 2008

predict the motion status of an object at some future moment [2]. They typically employ some kind of motion model to model the motion behavior of the objects for prediction. Motion synchronization methods consider network latency as an unavoidable problem in a networked environment and try to minimize the discrepancy of shared objects among the remote users. To do this, most synchronization methods also employ some kind of motion model to model the motion behaviors of the shared objects [3]. They first determine/approximate the network latency value and based on this delay value, they try to predict the actual position of the shared object.

We have been developing a game engine to support online gaming [4, 5]. The engine is based on dynamically transmitting geometry information to the client machines. The engine begins with a minimal amount of geometry information surrounding the location of the player. As the player moves around in the game scene, additional geometry information are sent to the client for viewing. Each object used in the engine is formatted in the form of a progressive mesh, which contains a base mesh and a sequence of progressive records. The difficulties of such an engine are not only to synchronize the interactions among multiple players but also to make sure that the visible geometry information are made available before they are needed. Both of them require a reliable motion predictor.

The rest of this note is outlined as follows. In Section 2, we review the two main types of motion prediction technique, short-term prediction and long-term prediction. In Section 3, we discuss our previous and current work on motion prediction for online gaming. We also present our current work in combining short-term prediction with long-term prediction for geometry prefetching. Finally, we briefly conclude this note in Section 4.

2 Related Work

There are many prediction techniques proposed for various applications. In this section, we focus our discussion on motion prediction methods. In general, motion prediction methods can be roughly classified into short-term prediction and long-term prediction. Short-term prediction refers to those prediction methods that are designed to model a very short term, typically with prediction length of less than one second, behavior of motion, while long-term prediction methods are designed to model a relatively longer term, typically over a few seconds or a minute, behavior of motion. In this section, we summarize existing work on these two types of motion prediction.

2.1 Short-Term Prediction

Majority of the methods developed for short-term motion prediction aim at addressing the network latency problem. One method is to acknowledge the existence of network latency by revealing the delay information explicitly to the users [6]. It is then up to the users to adjust their own actions to anticipate for the delay. Unfortunately, this method essentially passes the responsibility to the users to handle the latency problem themselves. The problem becomes even more severe when the latency fluctuates. Another method is to defer updating an object change until the client with the highest latency has received the update [7]. This, however, further increases the latency.

A more popular solution to network latency is dead reckoning [8, 9]. For each shared object, a motion predictor is run in the host machine as well as in all clients accessing the object. The host sends out an update message only when the error between the predicted state and the actual state of the object is higher than a given threshold. In between updates, the clients approximate the object state using the local motion predictor. Here, the polynomial predictor is often used as the motion predictor to extrapolate/predict future object locations. Although polynomials of any orders may be used, second order polynomials are most popular. Dead reckoning generally helps alleviate the network latency problem and also reduces the number of update messages needed to be sent through the network. However, its performance depends heavily on the accuracy of the predictor. In [4], an EWMA scheme is used in distributed virtual walkthrough. The model assigns different weights to past movement vectors, with higher weights to recent vectors.

In general, the above methods are designed for general motion prediction. There are a few specialized motion prediction methods proposed for predicting motion of specific objects. In [10], a predictor for head motion is proposed. It is based on the Kalman filter, which was originally used to filter measurement noise in linear systems by recursively minimizing the mean square estimation error. This method may work with the polynomial predictor or other prediction model to further reduce the prediction error. Results show that during rapid motion, the predictor becomes less effective and its performance is similar to one without using the Kalman filter. Another method is to use the Gauss-Markov process model to predict head motion [11] and human motion [12]. In [13], a grey system theory-based predictor, which accuracy is similar to the polynomial-based predictor with Kalman filtering, is used to predict head motion. Other than Kalman filtering, sequential Monte Carlo [14] has also been used in motion tracking. Although this method can approximate arbitrary distribution functions without uni-model Gaussian assumption, it is relatively expensive computationally. In [15], a double exponential smoothing-based predictor is proposed for predicting head and hand motion. It is efficient despite its slightly lower accuracy.

In general, specialized prediction methods produce more accurate results compared with general prediction methods and are more capable of predicting further ahead in time even if the motion is vigorous. This is because they take into account the motion behavior of the target objects. The tradeoff is the loss of generality.

2.2 Long-Term Prediction

Methods developed for long-term motion prediction are mainly for mobile networking or path planning. In mobile networking, motion prediction methods are used to predict the movement of mobile users in order to provide a smoother handover as the users move across zones or to improve bandwidth allocation among neighboring zones. An earlier approach to this considered human navigation as a random process in order to predict the next zone that a mobile user would likely visit. The area of each zone is typically in terms of square kilometers and the time staying in a zone is typically more than one minute. One of the mobility models proposed assumes that the traveling direction and velocity are uniformly distributed over a range and independent of each other [16]. Once the direction and velocity are computed, they are not changed. A revised method is to re-compute the direction and velocity after sometime [17] or some distance traveled [18]. In [19], a Gaussian-based random walk model is proposed. [20] extends the random walk model by assuming that the movement is fast and in a straight line and the occurrence of such movement is modeled by a Poisson process. In general, methods of this approach assume that the movement of the mobile users conforms to a random motion behavior. Unfortunately, this is rarely true and these methods may not be able to reflect the actual movement characteristics of the users. To model the movement behaviour of mobile users more accurately, a better approach is to use discrete state conditional probability. Given the previous and the current zones of a user, the probabilities of the user moving to different neighboring zones are evaluated from past history [21, 22]. [23] suggests considering the road topology to obtain a more accurate prediction. This approach has the advantage that the statistical information stored in the zones can be updated independently. However, it only considers the probability of accessing each neighboring zone.

While the above methods are mainly proposed to handle the handover of mobile users, there are methods that are proposed to handle bandwidth allocations. One approach is to record the movement history of each user and use it to estimate future trajectories. These methods consider the fact that most people tend to take the same paths when attending some regular activities, such as going from home to office. In [24, 25, 26], methods are proposed to periodically (e.g., once every hour) record which zone each user is visiting. The collected information can then be used to evaluate the probability of a user being in a particular zone at a particular period of time. However, this approach ignores the fact that this probability value is often dependent on the previous zones that the user has visited. Another approach is to store the exact paths (or the sequence of zones) in the database that the users have visited and then use pattern matching techniques to select similar paths. Given an ordered sequence of zones that a user has visited, the system may retrieve from the database the paths that are most similar to the visited path. The system may then predict future movement of the user based on the retrieved paths [27, 28]. To speed up the retrieval process, [29] proposes a method to reduce the number of paths needed to be examined by considering the user's current velocity and travel direction. In general, this approach may be able to estimate path probabilities. However, it fails if there are no similar paths available in the database.

In path planning, the objective is to determine an optimal path to move from one place to another while satisfying some predefined constraints [30]. Moving optimally may imply minimizing the distance traveled or effort spent. Predefined constraints can be the requirement to satisfy some physical limits, such as maximum velocity, or to avoid some collisions with obstacles. The problem can be formulated by constructing a connected graph, (V, E) where V is the set of vertices and E is the set of edges, with nodes covering the whole environment. Each edge is assigned a weight or cost value. By applying dynamic programming or the Dijkstra algorithm, the optimal path can be found. To determine the optimal path that avoids collisions, the probability of collisions in each zone can be evaluated through prior knowledge [31] or statistics from samples [32]. Instead of dividing the environments into zones to minimize collisions, [33] proposes to evade moving obstacles by considering their possible trajectories. It first records the paths of the moving obstacles and clusters similar paths. It then computes an optimal path that would avoid collisions with moving obstacles as much as possible. The limitation of this approach is that if the user's destination or an overall picture of the environment is not available, it would not be able to determine a suitable path.

3 Combining Short-Term and Long-Term Motion Prediction

3.1 Our Short-Term Prediction Approach

Our objective of a short-term prediction method for online gaming is to overcome the network latency. In general, network latency is typically in the order of 0.01s for local Internet connections and 0.2s for cross-country Internet connections. However, in situations where a client needs to receive a return message from the receiver, the delay will be at least double.

As specialized prediction methods are designed to model the motion behavior of the target object to achieve better prediction performance, they may not be easily adapted to model the motion behavior of other objects. We have noticed that in desktop 3D applications, such as virtual walkthrough and online gaming, the 2D mouse is still the most popular device used as navigation input. Through studying the motion behavior of a mouse during 3D navigation, we have developed an elliptic model to model the motion behavior of the 2D mouse during such navigation and proposed a hybrid motion prediction method to predict the mouse motion – a linear model for prediction at low velocity motion and the elliptic model for prediction at high velocity motion [2]. To map the 2D mouse motion into 3D motion, we decompose the 2D mouse motion into two time-series, x motion and y motion, and compute the predicted x motion and y motion in each frame. By mapping the predicted 2D velocity vector into the 3D scene, we may then predict the future position of the object in the 3D scene. The elliptic model is defined as follows:

$$v = K_1(\cos(K_2\, t) - 1) \tag{1}$$

where v is the predicted velocity at the predicted time t, since the start of the motion pulse. K_1 and K_2 are referred to as the pulse constants. They are the parameters of the motion pulse and are constants during the period of the pulse.

Figure 1 shows the prediction error of this hybrid motion prediction method (Phm) as compared with those of two popular predictors, the second order polynomial predictor (Psop) [34] and the Guass-Markov process model (Pgmm) [11]. In general, the prediction error increases exponentially as we increase the prediction length. However, the hybrid motion predictor has a much lower prediction error. With such favorable prediction accuracy, we may use it for motion synchronization among the remote players [3]. We may also use it for geometry prefetching [5] in our streaming-based game engine.

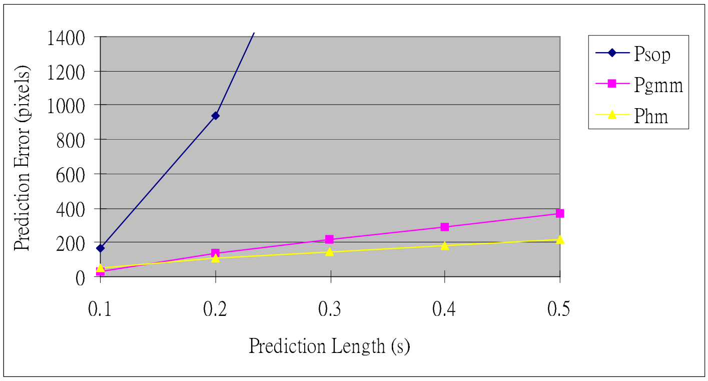


Fig. 1. Prediction error of the hybrid motion prediction method

3.2 Our Long-Term Prediction Approach

Our objective of a long-term prediction method for online gaming is more to help improve resource allocation and optimization, which typically take longer time than the network latency to complete the operation.

In long-term prediction, we are focusing our effort on developing a statistical method. It is by dividing the environment into zones (square cells). These zones form basic units for statistical prediction. We collect statistical movement data of players moving in and out of each zone. Given the location, x, where a player enters a zone, we may predict the probability that the player will leave the zone at a particular boundary location, y, based on the collected statistics as shown in Figure 2. We model such a probability distribution function $f(y|x)$ as a Gaussian mixture as follows:

$$f(y|x) = \sum_{i=1}^{n} a_i f_i(y;\ C_i,\ m_i) \qquad (2)$$

where n is the number of mixture. $f_i(y;\ C_i,\ m_i)$ is the i^{th} Gaussian distribution function with covariance C_i and mean m_i, $0 < a_i < 1$, and $\sum_{i=1}^{n} a_i = 1$. In other words, the distribution function is a weighted sum of several Gaussian distributions.

Based on the computed probability values, we may then estimate which neighboring zone(s) that a player will likely visit next. Figure 3 shows some initial results as we compare the prediction error of our long-term prediction method with those of two short-term prediction methods (the second order polynomial and the Guass-Markov process model) and a long-term prediction method [22]. From the results, our method seems to perform well compared with all the other methods. We can also see that long-term prediction methods generally produce high prediction accuracy at high prediction lengths.

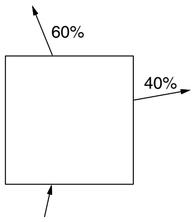


Fig. 2. Statistical prediction of exit-positions, depending on a given enter-position

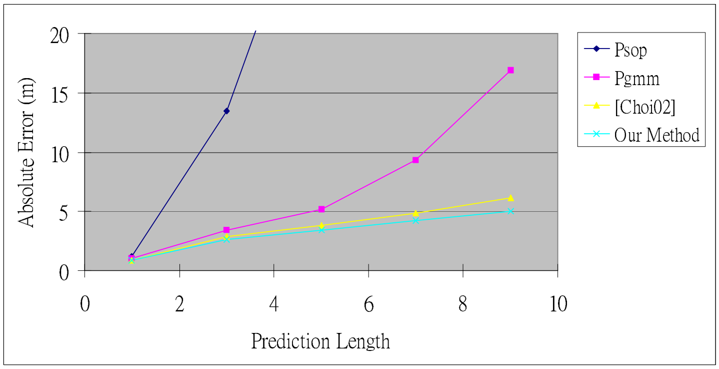


Fig. 3. Prediction error of the statistical long-term prediction method

Our new long-term prediction method can be very useful, as we may accumulate statistics from zone to zone to construct a path tree. This path tree shows us the probability of various paths reachable from the user's current position. However, this also leads to a problem of how we should measure the prediction error. With a single predicted path, we may simply compute the Euclidean distance of the user's actual location from the predicted location. With many potential paths each with a different probability, it is no longer a straightforward matter to compute the accuracy. For example, we assume that a user is currently in a zone that shows an exit to the left with a 60% probability and an exit to the right with a 40% probability. If the user ends up turning left, should we say that the prediction has a 100% accuracy or 60% accuracy? What if the user decides to move upward (instead of left or right)? The situation becomes more complex as we accumulate the statistics from zone to zone. In Figure 3, we consider only the exit with the highest probability when we measured the error of our method.

In Figure 4, we show an example path tree. The left image shows an aerial view of part of a 3D environment of the Central London, while the right image shows the path tree superimposed on the aerial image. The black square near to the center of the image represents the zone where the user is currently located and the other grey zones indicate the probabilities that the user may move into the zones in the near future (with darker zones having high probability values and lighter zones having low probability values).

Currently, we are also investigating two applications of this long-term prediction method. One is resource allocation and optimization. For example, in a multi-server environment [35], if we know that a large amount of players will move to a particular zone at a particular future moment, we may initiate a load balancing process to address the potential sudden increase in workload of the zone well before the zone becomes overloaded. Another application of the long-term prediction method is geometry prefetching. If we know that a player will likely move to a particular neighboring zone, wemay prefetch objects in the zone to the client in advance. With a long-term prediction method, this will allow us to have much more time to prefetch objects. However, due to the relatively high prediction error at high prediction lengths, it is possible that a lot of bandwidth can be wasted prefetching objects that will never be used.

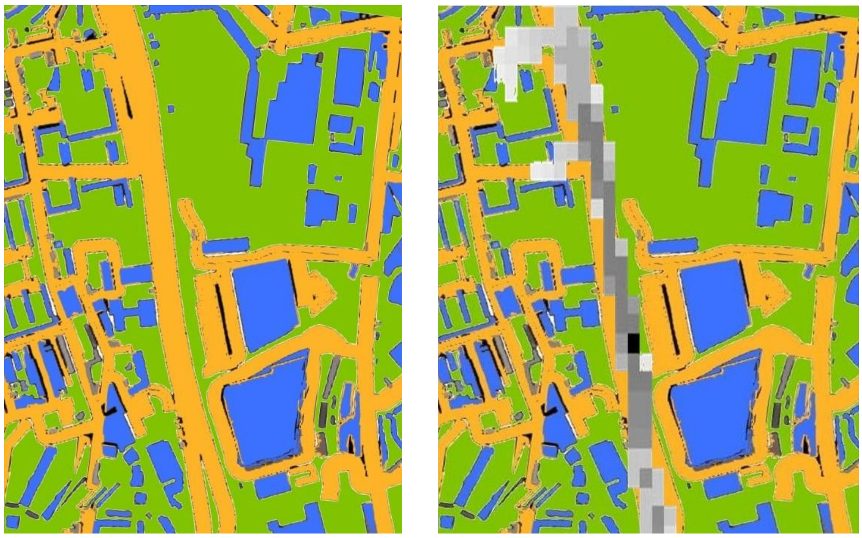


Fig. 4. An example path tree: left image shows part of the Central London while the right image shows a superimposed path tree, starting from the user's position (the black zone)

3.3 A Combined Motion Prediction Approach

In general, our short-term motion predictor returns a precise predicted location, given a prediction length. It also produces high prediction accuracy when the prediction length is below one second. However, when applied to the streaming-based game engine for object prefetching, its performance can be significantly affected by a few factors, such as the amount of geometry information needed to be sent, the available network bandwidth, the CPU processing delay, and the actual network latency. This is because the delay caused by these factors is comparable with the prediction length of a short-term predictor. For example, a higher network latency or CPU processing delay may significantly reduce the amount of time left for object prefetching. A region with a lot of high resolution objects may also consume a significant percentage of the network bandwidth for a period of time.

On the other hand, the long-term motion predictor that we are developing returns much less precise location information. It only indicates possible regions where the user may be moving to in the future. At the same time, the prediction accuracy is also less reliable, compared to that of the short-term predictor at low prediction length. However, the long-term motion predictor can offer a much higher prediction length, i.e., can predict much further ahead in time. Hence, when applied to the streaming-based game engine for object prefetching, it provides us more time to prefetch objects. However, because the predictor returns much wider regions that the user may be moving to, prefetching all the potentially visible objects in these regions can be very expensive and may also use up a large portion of the network bandwidth.

A major objective for us to develop a predictor that combines both short-term prediction and long-term prediction is to improve the prefetching performance of the streaming-based game engine. One method that we are considering is to use the predicted results from the long-term predictor to prefetch objects at some coarse resolution, e.g., base meshes, while those from the short-term predictor to prefetch progressive records to refine the prefetched objects. The idea of this is that it is more important to be able to show at least a low resolution model of an object than missing it when needed. Hence, the long-term predictor is to make sure that those potentially visible objects, although in large quantity, are made available in the client machine at some low resolution to minimize the bandwidth consumption, while the short-term predictor is to try to obtain enough progressive records for each object to optimize the visual quality. Here, we will need to investigate on how to determine the appropriate resolutions based on the predicted results and perhaps also on some dynamic visual factors such as object/zone distance from the player.

4 Conclusion

In this note, we have discussed existing work and our work on both short-term and long-term motion prediction. We have discussed their features and how they may be used to help prefetch object in our streaming-based game engine.

From our experience, the prediction error of most motion prediction methods, whether they are long-term or short-term, is an exponential function of the prediction length. In general, the prediction error is roughly linear at the beginning as we increase the prediction length. However, after some critical point (referred to as the *critical prediction length*), the prediction error increases significant. We have found that it is the linear region (before the critical prediction length) which is most useful for prediction, as it provides a predictable performance.

Currently, we are investigating the possibility of using the prediction results from one predictor to improve the prediction results of the other. We are looking at how to make use of the long-term prediction results to improve the prediction accuracy of the short-term motion predictor and vice versa.

Acknowledgements

We would like to thank Anthony Steed for providing the 3D geometry data of the Central London. The work described in this paper was partially supported by a SRG grant from City University of Hong Kong (Project Number: 7002309).

References

1. Final Fantasy XI, `http://www.playonline.com/ff11us/`
2. Chan, A., Lau, R., Ng, B.: Motion Prediction for Caching and Prefetching in Mouse-Driven DVE Navigation. ACM Trans. on Internet Technology 5(1), 70–91 (2005)
3. Li, L., Li, F., Lau, R.: A Trajectory-Preserving Synchronization Method for Collaborative Visualization. IEEE Trans. on Visualization and Computer Graphics 12(5), 989–996 (2006)
4. Chim, J., Green, M., Lau, R., Si, A., Leong, H.: On Caching and Prefetching of Virtual Objects in Distributed Virtual Environments. In: Proc. ACM Multimedia, pp. 171–180 (September 1998)
5. Li, F., Lau, R., Kilis, D.: GameOD: An Internet Based Game-On-Demand Framework. In: Proc. ACM VRST, pp. 129–136 (November 2004)
6. Vaghi, I., Greenhalgh, C., Benford, S.: Coping with Inconsistency due to Network Delays in Collaborative Virtual Environments. In: Proc. ACM VRST, pp. 42–49 (1999)
7. Mauve, M., Vogel, J., Hilt, V., Effelsberg, W.: Local-lag and Timpwarp: Providing Consistency for Replicated Continuous Applications. IEEE Trans. on Multimedia 6(1), 47–57 (2004)
8. DIS Steering Committee, IEEE Standard for Distributed Interactive Simulation - Application Protocols, IEEE Standard 1278 (1998)
9. Singhal, S., Zyda, M.: Networked Virtual Environements: Design and Implementation. ACM Press, New York (1999)
10. Azuma, R., Bishop, G.: A Frequency-Domain Analysis of Head-Motion Prediction. In: Proc. ACM SIGGRAPH, pp. 401–408 (1995)
11. Liang, J., Shaw, C., Green, M.: On Temporal-Spatial Realism in the Virtual Reality Environment. In: Proc. ACM UIST, pp. 19–25 (1991)
12. Capin, T., Pandzic, I., Magnenat-Thalmann, N., Thalmann, D.: A Dead-Reackoning Algorithm for Virtual Human Figures. In: Proc. IEEE VRAIS, pp. 161–169 (1997)
13. Wu, J., Ouhyoung, M.: On Latency Compensation and its Effects on Head-motion Trajectories in Virtual Environments. The Visual Computer 16(2), 79–90 (2000)
14. Isard, M., Blake, A.: CONDENSATION – Conditional Density Propagation for Visual Tracking. Int'l Journal of Computer Vision 29(1), 5–28 (1998)
15. LaViola Jr., J.: An Experiment Comparing Double Exponential Smoothing and Kalman Filter-Based Predictive Tracking Algorithms. In: Proc. IEEE VR, pp. 283–284 (2003)
16. Thomas, R., Gilbert, H., Mazziotto, G.: Influence of the Movement of Mobile Station on the Performance of the Radio Cellular Network. In: Proc. 3rd Nordic Seminar, paper 9.4 (1988)
17. Xie, H., Goodman, D.: Mobility Models and Biased Sampling Problem. In: Proc. IEEE ICUPC, pp. 803–807 (1993)
18. Broch, J., Maltz, D.A., Johnson, D., Hu, Y., Jetcheva, J.: A Performance Comparison of Multi-Hop Wireless and Ad Hoc Network Routing Protocols. In: Proc. ACM MOBICOM, pp. 85–97 (1998)

19. Rose, C.: Minimizing the Average Cost of Paging and Registration: A Timer-Based Method. Wireless Networks 2(2), 109–116 (1996)
20. Brown, T., Mohan, S.: Mobility Management for Personal Communication Systems. IEEE Trans. on Vehicular Technology 46(2), 269–278 (1997)
21. Bar-Noy, A., Kessler, I., Sidi, M.: Mobile Users: To Update or not to Update? Wireless Networks 1(2), 175–185 (1995)
22. Choi, S., Shin, K.: Adaptive Bandwidth Reservation and Admission Control in QoS-Sensitive Cellular Networks. IEEE Trans. on Parallel and Distributed Systems 13(9), 882–897 (2002)
23. Soh, W., Kim, H.: A Predictive Bandwidth Reservation Scheme Using Mobile Positioning and Road Topology Information. IEEE/ACM Trans. on Networking 14(5), 1078–1091 (2006)
24. Tabbane, S.: An Alternative Strategy for Location Tracking. IEEE Journal on Selected Area in Communication 13(5), 880–892 (1995)
25. Bhattacharya, A., Das, S.: LeZi-Update: An Information-Theoretic Framework for Personal Mobility Tracking in PCS Networks. Wireless Networks 8(2-3), 121–135 (2002)
26. Shen, X., Mark, J., Ye, J.: User Mobility Profile Prediction: An Adaptive Fuzzy Inference Approach. Wireless Networks 6(5), 363–374 (2000)
27. Liu, G., Maguire Jr, G.: A Class of Mobile Motion Prediction Algorithms for Wireless Mobilecomputing and Communications. Mobile Networks and Applications 1(2), 113–121 (1996)
28. Liu, T., Bahl, P., Chlamtac, I.: Mobility Modeling, Location Tracking, and Trajectory Prediction in Wireless ATM Networks. IEEE Journal on Selected Area in Communication 16(6), 922–936 (1998)
29. Akyildiz, I., Wang, W.: The Predictive User Mobility Profile Framework for Wireless Multimedia Networks. IEEE Trans. on Networking 12(6), 1021–1035 (2004)
30. LaValle, S.: Planning Algorithms. Cambridge Press (2006)
31. Tado, S., Hayashi, M., Manabe, Y.: Motion Planner of Mobile Robots which Avoid Moving Human Obstacles on Basis of Stochastic Prediction. In: Proc. IEEE SMC, pp. 3286–3291 (1995)
32. Tana, K.: Detecting Collision-Free Paths by Observing Walking People. In: Proc. IEEE Intelligent Robots and Systems, pp. 55–60 (2002)
33. Kruse, E., Gutsche, R., Wahl, F.: Acquisition of Statistical Motion Patterns in Dynamics Environments and their Application to Mobile Robot Motion Planning. In: Proc. IEEE Intelligent Robots and Systems, pp. 712–717 (1997)
34. Cai, W., Lee, F., Chen, L.: An Auto-adaptive Dead Reckoning Algorithm for Distributed Interactive Simulation. In: Proc. Workshop on Parallel and Distributed Simulation, pp. 82–89 (1999)
35. Ng, B., Si, A., Lau, R., Li, F.: A Multi-Server Architecture for Distributed Virtual Walkthrough. In: Proc. ACM VRST, pp. 163–170 (November 2002)

Two-Character Motion Control: Challenge and Promise

Sung Yong Shin

Division of Computer Science, KAIST
syshin@jupiter.kaist.ac.kr

Abstract. The problem of two-character motion control is important, but has drawn limited attention from computer animation research community. This paper is intended to highlight this problem to the research community to attract more attention. We explore research issues in two-character motion control and discuss our experiences with emphasis on applications such as coupled dancing and kickboxing.

1 Background

For the last decade, data-driven character animation has been gaining ever-increasing popularity. From the motion control point of view, however, the main stream of research has still been on single-character control, in which little attention is paid to interactions between characters.

Two-character interactions are observed as an integral part of our daily life. These interactions drive a variety of two-charactger motions, for example, ball games such as tennis and badminton, martial arts such as kickboxing, Karate, and Taekwon-do, fighting sports such as boxing and wrestling, coupled dances such as tango and waltz, and so on, to name a few. Two-character motions are also observed in computer games and animations. However, tow-character motion control has drawn limitd attention from the computer animation research community.

In this paper, we aim at exposing this important research problem to the computer animation research community by closing up the issues in it while also reporting our experiences in these issues, with emphasis on coupled dancing and kickboxing.

The remainder of the paper is organized as follows. In Section 2, we review previous work. Then, we raise issues in two-character motion control in Section 3. In Section 4, we discuss our experiences and intuition. We conclude the paper in Section 5.

2 Related Work

A characteristic feature of two-character motion control is how to deal with mutual interactions. A primitive form of interactions arises by physical contacts.

© Springer-Verlag Berlin Heidelberg 2008

Zordan and Hodgins[3] synthesized reactive upper-body motions to various impacts by the opponent in boxing and table tennis, by incorporating physical simulation into data-driven animation. Zordan et al.[4] later extended this method for full-body motion. For reactive motions to pushing, Arikan[5] presented a method for identifying realistic motions. Although being able to synthesizing reactive motions to external forces exerted by physical contacts between characters, the above methods did not address their mutual interactions.

Kim et al.[1] presented a data-driven method to rhythmic motion synthesis. They demonstrated an impressive animated scene of ballroom dancing by multiple couples, each being regarded as a single entity. The motion stream of each couple was synthesized in accordance with background music while traversing a motion transition graph constructed from captured motion data. Hsu et al.[2] provided a method to synthesize a stream of coupled dancing motions. The motion of a dancing character is used as the control signal to search for the corresponding motion of the partner from motion databases. The premise for both methods is that motions of a dancing pair are tightly coupled in both time and space.

Liu et al.[6] proposed a physics-based method to generate multi-character motions from short single-character motion clips. A space-time optimization is formulated to find multi-character motions with the constraints that reflect the desired character interactions. This formulation is intended for off-line applications.

Coupled hidden Markov models (CHMMs) [7] [8] have been commonly adopted to capture cross dependencies between two or more synchronous signals such as audio/visual signals[9]. A CHMM models each signal with a set of states and their transitions. The state transitions for each signal are synchronized with those of the others.

Recently, Kwon et al[10] presented a data-driven method for synthesizing standing-up martial arts such as kickboxing, Taekwondo, and Karate performed by a pair of characters. Adopting a dynamic Baysian network, the authors were able to model asynchronous motion transitions while reflecting character interactions captured from example motion data.

3 Problem Statement

Our objective for two-character motion control is to support real-time applications including video games. Thus a solution should satisfy the following requirements:

- On-line real-time performance
- Synthesizing believable motions
- Being flexible enough to accommodate a variety of scenarios (possibly not preplanned)

These requirements can best be satisfied by data-driven approaches.

A popular data-driven paradigm for video game creation is first to preplan a scenario, and then to capture and edit motion data according to the preplanned scenario. In runtime, captured motion clips are replayed in response to requests by the users, guided by the scenario. The problem with this paradigm arises mainly due to dependency of motion data on scenarios.

Two-character motion control deals with two (possibly different) motions simultaneously performed by a pair of characters. At every instant of time, the poses of two characters are specified by at least twice as many degrees of freedom as the pose of a single character. If two-character motions were simultaneously considered, a large amount of motion data would be required to cope with the large dimensionality of the two-character motion space.

Adopting a data-driven approach, we need to address the following issues :

- How to model two-character motions effectively with a limited amount of motion data
- How to model the interactions between characters
- How to synthesize a wide range of believable motions in an on-line real-time manner so as to support various scenarios

The first two issues are related to motion analysis while the last issue concerns motion synthesis.

4 Approaches

In this section, we discuss our experiences by considering the solutions to two extreme cases, coupled dancing and kickboxing. The both solutions share the same framework as shown in Figure 1. This framework consists of two parts: analysis and synthesis. The analysis part addresses two issues, motion modeling and interaction modeling, and the synthesis part addresses the motion synthesis issue.

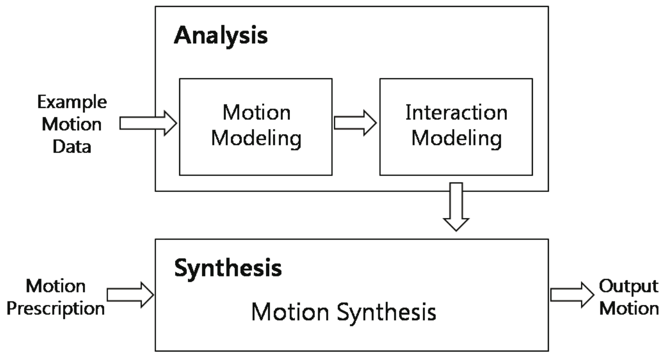


Fig. 1. Solution Framework

4.1 Synchronous Case : Coupled Dancing

In coupled dancing, a man leads a woman follower, and the woman tries to follow the man's lead to tightly couple their motions, guided by a piece of background music. Furthermore, the couple moves in accordance with dancing rules, which limit their moving patterns. Thus, a motion of either person can be used as a control signal for that of the other. Kim et al.[1] exploited this observation to present an interesting approach to coupled dancing motion synthesis as shown in Figure 2.

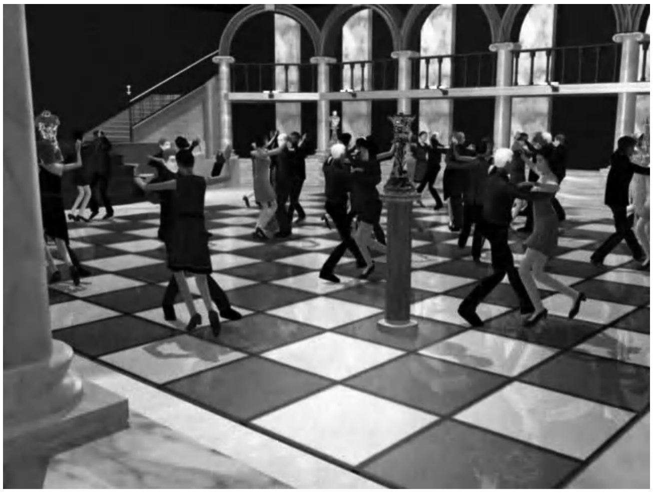


Fig. 2. Coupled dancing

In particular, only the motion stream of one person is used for motion analysis while ignoring the other.

Motion modeling consists of three steps: motion segmentation, motion classification and motion transition graph construction. The approach of Kim et al.[1] first cuts an example coupled dancing motion stream into short motion segments based on the beat patterns embedded in the stream. A motion transition graph is then constructed using only single person's motion segments, where a node represents a set of motion segments with the similar logical structure and an edge represents motion transition. K-means clustering is used to classify motion into the nodes, and the transition probabilities are learned from the example motion data based on a Markov random field model. The approach skips interaction modeling since a motion of a person implies that of the other. For motion synthesis, the motion transition graph is traversed from node to node in accordance with music guided by transition probabilities while generating a single person's motion segment at each node. This motion segment is used a control signal to determine the partner's corresponding motion segment.

4.2 Asynchronous Case : Kickboxing

Unlike coupled dancing, kickboxing is quite asynchronous. A player tries to fool the opponent to seek a chance for an attack or a counterattack. Such an attempt is likely to be most successful when one player moves against the other's expectation in both time and space. Thus, a motion of either player may not be used as a control signal. Kwon et al.[10] presented a framework for two-character motion analysis and synthesis with focus on martial arts such as kickboxing as shown in Figure 3.

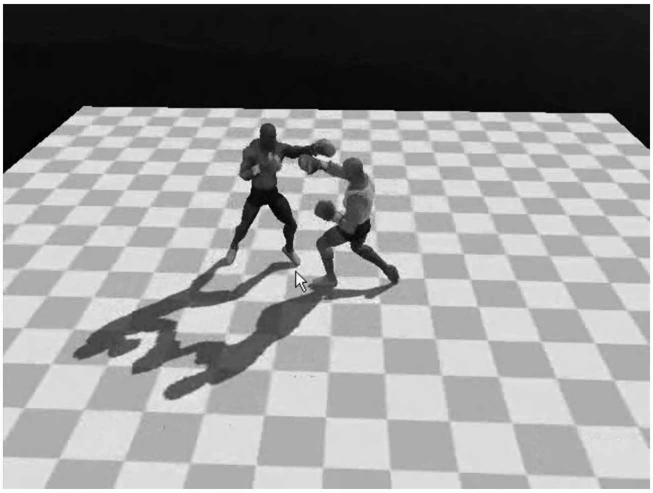


Fig. 3. Kickboxing

For motion modeling, we also adopt a three-step approach as for coupled dancing. Our final goal is to construct a coupled motion transition graph by combining a pair of motion transition graphs for the players with a set of cross edges. Every cross edge connects a pair of nodes from different motion transition graphs to capture an interaction between the players. In order to build each player's motion transition graph, the example two-player motion stream is decoupled into a pair of individual player's motion streams. Each individual player's motion stream is separately segmented and classified into a collection of motion groups, each of which corresponds to a node of an individual player's graph. The contact force profile of the individual motion stream is used for motion segmentation, and multiclass support vector machine classifiers are used for motion classification.

For interaction modeling, a dynamic Bayesian network is employed to capture the causal relationships of coupled actions embedded in the example motion stream. Specifically, the next action of a player depends on the current actions of both players. These motion transition patterns result from interactions between the players, which are used as building blocks to obtain a dynamic Bayesian network. In addition, the contact instants between the players are also captured to further enhance synthesized motion quality.

For motion synthesis, a pair of characters exchange actions and reactions via the cross edges while traversing their respective single-player motion transition graphs, guided by the dynamic Bayesian network. One of the characters may be designated as an avatar. This character is controlled by the user in an online manner. The dynamic Bayesian network is extended to incorporate the control information such as action prescriptions by the user.

5 Conclusions

Two-character motion control has not yet drawn deserved attention from the computer animation research community. Although partial solutions were proposed for simple applications, a general solution is far beyond our reach. This paper is intended to expose the challenging and yet important research problem while also showing promise by highlighting existing approaches.

Acknowledgements. This work was parly supported by the Korea Research Foundation, under grant no.D00190.

References

1. Kim, T.H., Park, S.I., Shin, S.Y.: Rhythmic-Motion Synthesis Based on Motion Beat Analysis. ACM Trans. Graphics 22(3) (2003); Proc. ACM SIGGRAPH 2003, pp. 392–401 (July 2003)
2. Hsu, E., Gentry, S., Popovic, J.: Example-Based Control of Human Motion. In: Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA 2004), pp. 69–77 (2004)
3. Zordan, V.B., Hodgins, J.K.: Motion Capture-Driven Simulations that Hit and React. In: Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA 2002), pp. 89–96 (2002)
4. Zordan, V.B., Majkowska, A., Chiu, B., Fast, M.: Dynamic Response for Motion Capture Animation. ACM Trans. Graphics 24(3), 697–701 (2005)
5. Arikan, O., Forsyth, D.A., O'Brien, J.F.: Pushing People Around. In: Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA 2005), pp. 56–66 (July 2005)
6. Liu, C.K., Hertzmann, A., Popovic, Z.: Composition ofComplex Optimal Multi-Character Motions. In: Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA 2006), pp. 215–222 (2006)
7. Brand, M.: Coupled Hidden Markov Models for Modeling Interacting Processes, Technical Report 405, MIT Media Lab (1996), `http://xenia.media.mit.edu/brand/Publications.html`
8. Brand, M., Oliver, N., Pentland, A.: Coupled Hidden Markov Models for Complex Action Recognition. In: Proc. IEEE CS Conf. Computer Vision and Pattern Recognition, pp. 994–999 (1997)
9. Chu, S.M., Huang, T.S.: Audio-Visual Speech Modeling Using Coupled Hidden Markov Models. In: Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing, pp. 2009–2012 (2002)
10. Kwon, T., Cho, Y.S., Park, S.I., Shin, S.Y.: Two-character Motion Analysis and Synthesis. IEEE TVCG 14(3), 707–720 (2008)

Motion Modeling: Can We Get Rid of Motion Capture?

Daniel Thalmann

EPFL Vrlab
Station 14
CH 1015 Lausanne, Switzerland
{Daniel.Thalmann}@epfl.ch

Abstract. For situations like crowd simulation, serious games, and VR-based training, flexible and spontaneous movements are extremely important. Motion models would be the best strategy to adopt, but unfortunately, they are very costly to develop and the results are disappointing. Motion capture is still the most popular way. The ultimate in terms of motion models seems to be data-driven. Motion retargeting and PCA-based models are well used but they still rely strongly to Motion Capture. In this paper, we try to analyze the situation and illustrate it using a few case studies.

Keywords: Motion Capture, Motion Editing, Motion Models.

1 Introduction

Traditionally, Computer Graphics has been arbitrarily divided into three main tracks: Geometric modeling, rendering, and animation. Looking at descriptions of Computer Graphics Courses, we may see very often descriptions mentioning: modeling techniques including curves and surfaces, reflection models and illumination algorithms, and basic methods for animation. This summarizes the trend: there are models for creating surface-based objects, models for rendering, but very few models for animation. If we consider the creation of 3D objects, we may easily use as a source the complete field of geometric modeling. In terms of rendering, the number of available models is almost unlimited: from illumination models like Phong's model to ray tracing, radiosity, and point-based models. Unfortunately, the situation is completely different for Computer Animation. Models are well used for physics-based animation like cloth animation for example. However, pure motion models are not common for character animation and when they exist, they are generally not used, because of the bad results they produce. Treuille et al. [1] state that despite decades of advances, interactive character animation still lacks the fluidity and variability of real human motion. Capturing the fine nuances of human motion requires a high-dimensional motion repertoire and a multivariate space of input parameters which can change continuously, often unpredictably. To model such fine nuances is just out of scope.

Let us consider the most popular motion: locomotion. Multon et al [2] have proposed a survey of methods to create walking models. They first review procedural methods based on knowledge-based kinematic animation. Then, they describe attempts to incorporate dynamic constraints in the generation of motion or to use

© Springer-Verlag Berlin Heidelberg 2008

dynamic simulation. Lastly, they present approaches enabling the interactive edition of either captured or synthetic walking motions. This means that the ultimate to model walking motion seems to be data-driven. Controllers exist, but walking style is dependent on the human size, the human weight, the mood, bags, or type of shoes. No current model takes into account all these features.

And what is used in movies or even games ? The most common way of creating animation sequences is still performance animation or motion capture which consists of measurement and recording of direct actions of a real person or animal for immediate or delayed analysis and playback. But the use of performance animation has enormous disadvantages:

- Specific hardware and special programs are required to obtain and process the data
- Cost of software and equipment and personnel required can be prohibitive
- Motion capture systems may have specific requirements for the space in which they are operated
- Applying motion capture to quadruped characters can be difficult
- Technology can become obsolete every few years as better software and techniques are invented
- Results are limited to what can be performed within the capture volume without extra editing of the data
- Movement that does not follow the laws of physics generally cannot be represented
- If the computer model has different proportions from the capture subject, artifacts may occur. For example, if a cartoon character has large, oversized hands, these may intersect strangely with any other body part when the human actor brings them too close to his body. In the same way, feet may enter the floor if characters' legs are longer than the human actors' legs
- Motion capture does not bring any really new concept to animation methodology. For any new motion, it is necessary to record the reality again. Moreover, motion capture is not appropriate, especially in real-time simulation activities, where the situation and actions of people cannot be predicted ahead of time, and in dangerous situations, where one cannot involve a human actor

But, finally the main disadvantage is that it is impossible to use it for animation of large groups of people like crowds, where we want to have tens of thousands of characters with an individual style of walking.

In the research community, motion capture is also very popular. If we consider papers on character animation in the Proceedings of SIGGRAPH, SCA, and CASA in 2006 and 2007, we find 37 papers. 12 are pure motion capture papers, 19 are strongly related to motion capture, 3 present motion controllers based on motion capture, and only 3 are pure controllers free from motion capture. This is disturbing, when we think that few papers on shapes are based on scanners and few papers on rendering need light capturing.

When motion capture is not used, people still consider key-frame animation. But, real-time systems must react fast enough to events, to keep a good immersion. This is particularly true when animation should "stick" to sound events, which is deeply affected by any delay in the processing. For example, in the case of virtual orchestra, we need to synchronize the musician's animation with the corresponding sound. Due to real-time restrictions it's impossible to predict the transition of states, and thus we cannot prepare the right key-frame at a certain time.

2 Case Studies at VRlab

In this Section, we review a few experiences in our Lab, showing the difficulties to generate flexible and spontaneous motion sequences.

2.1 Avatar against Agent

In this experience, Emering et al. [3] produced a fighting between a real person and an autonomous actor. The motion of the real person was captured using a magnetic Motion Capture system. The gestures are recognised by the system and the information is transmitted to the agent who is able to react to the gestures and decide which attitude to do. Figure 1. shows a snapshot of a life participant with ten sensors used to reconstruct the avatar in the virtual scene. The participant performs fight gestures which are recognized by the virtual opponent, a pure agent. The latter responds by playing back a pre recorded animation sequence. Because of the lack of a model and a too small motion database, the actions of the agent were repetitive and not always smooth enough.

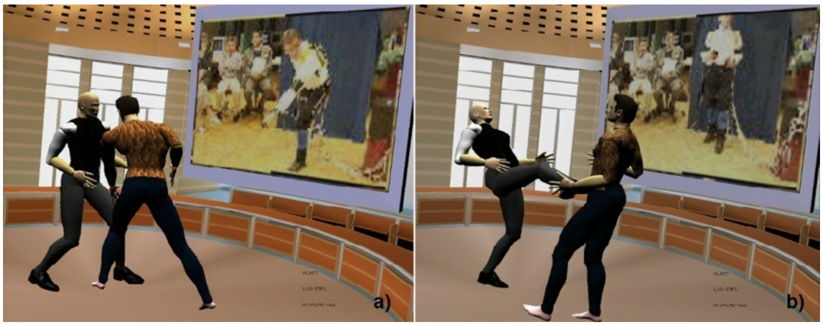


Fig. 1. Action Recognition for the avatar and motion synthesized for the agent

2.2 Training through Virtual Humans

Let us consider now a scenario that we implemented for the need of a medical European project dedicated to the training of paramedical personnel to emergency rescue [4]. The aim of this scenario was to train the Basic Life Support medical procedure. The trainee was immersed in a virtual office and discovered a man lying on the ground. He had to give BLS to the victim by giving orders to a young Virtual Assistant (VA) (Fig.2). The VA possesses all the skills required (mouth-to-mouth, chest compression) but highly hesitated on the procedure order; this was the job of the trainee. The latter interacted with the VA to apply his/her theoretical knowledge of

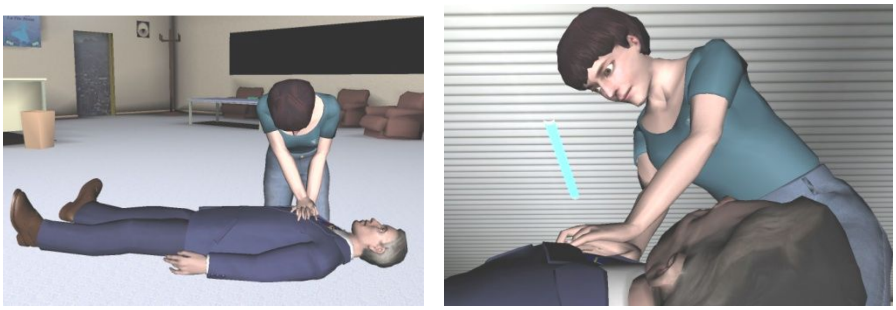


Fig. 2. Use of a Virtual Assistant for Basic Life Support Training

the BLS procedure. This scenario consisted of a dozen steps. At each step, the trainee had to make a (fast) decision. It does not present many possibilities because the BLS guidelines gave no alternative.

In many situations, autonomous agents should work as a team with the user to solve the problem at hand. In general, there was a plenty of work done on both multi-agent systems (mostly AI) and collaborative VR (mostly multi-user systems), but nobody tried to use collaborative agents for VR applications and have them collaborate with the human operator to solve a problem. This approach has a large potential for training applications (emergency response, medical training, air traffic control, even military) or for simulations (crowd control for example).

Again, the main restriction in such an approach is the lack of models to achieve tasks. A typical Virtual Character will only be able to play sequences (motion capture or key-frame based sequences).

2.3 Reflex Movements

In [5], we developed an observation-based approach to stereotype reaction movements of the humans. Our case study consisted in a simple reaction movement: make people react to a ball thrown towards them (Fig.3). Reactions to this kind of situations vary in many different ways from one person to another; they also vary over time, it means that the same person will react differently to the same stimulus. This variation depends on the personality and internal state of each person at that moment. The simplest spontaneous behavior in human beings produced by external stimuli is the reflex. Reflexes are unconscious movements and are regulated by the nervous system. They can be considered as the basis of movements or even as the root of human movement.

To synthesize movements, we have used the Inverse Kinematics library developed in [6]. This tool allows controlling more than one end-effector at the same time. It is possible to associate a distinct priority level to each effector to guarantee that the most important goals are achieved first. It also allows setting an engagement of the end effector joint with its parent joints in different levels. Moreover, we can treat the center of mass of the body as an end-effector to ensure the static equilibrium of a figure[7].

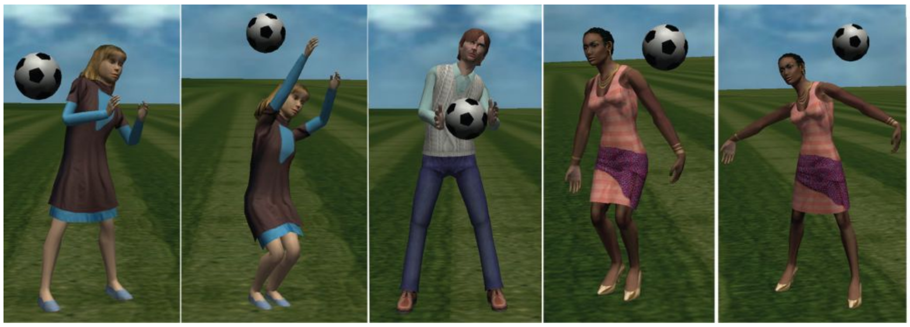


Fig. 3. Reflex Movements

Combining effectors and their associated weight and priority levels, we can simulate complex and believable synergistic postures, but this is still hard to make them believable unless we use motion capture sequences, but this prevents us to change the conditions like the type of ball for example.

3 Does Physics Solve the Problem?

Dynamics approaches aim to describe a motion by applying physics laws. For example, it is possible to use control algorithms based on finite state machine to describe a particular motion and proportional derivative servos to compute the forces [8]. However, even if these methods produce physically correct animations, the configuration of their algorithms remains difficult. It is not easy to determine the influence of each parameter on the resulting motions. Many methods based on empirical data and biomechanical observations are able to generate walking [9] or running patterns [10], reactive to given user parameters. Other similar approaches take into account the environment, to walk on uneven or sloped terrains [11] or to climb stairs [12]. Despite their real-time capability, all these methods lack realism, as the legs' motion is considered symmetrical, for example.

4 Motion Editing

Motion retargeting or motion editing is simply editing existing motions to achieve the desired effect. Since motion is difficult to create from scratch using traditional methods, changing existing motions is a quicker way to obtain the goal motion. This means that motion editing is a well known approach used to produce new motions from existing ones. Constrained based techniques enable the user to specify constraints over the entire motion or at specific times while editing an animation. Most approaches [13, 14, 15] solve the synthesis problem on the full-postural space described by the characters joint angles. The problem of low-dimensional human motion synthesis has been already addressed by some authors [16, 17, 18, 19].

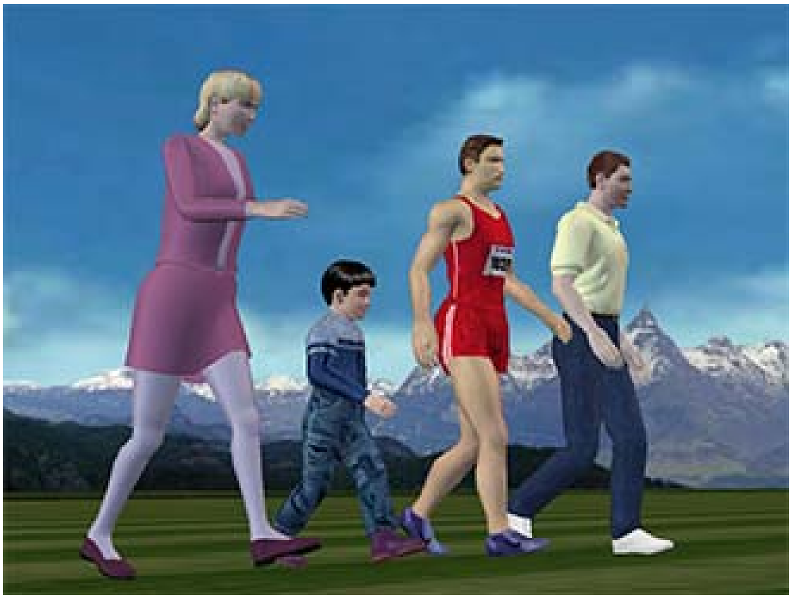


Fig. 4. PCA-based walking models

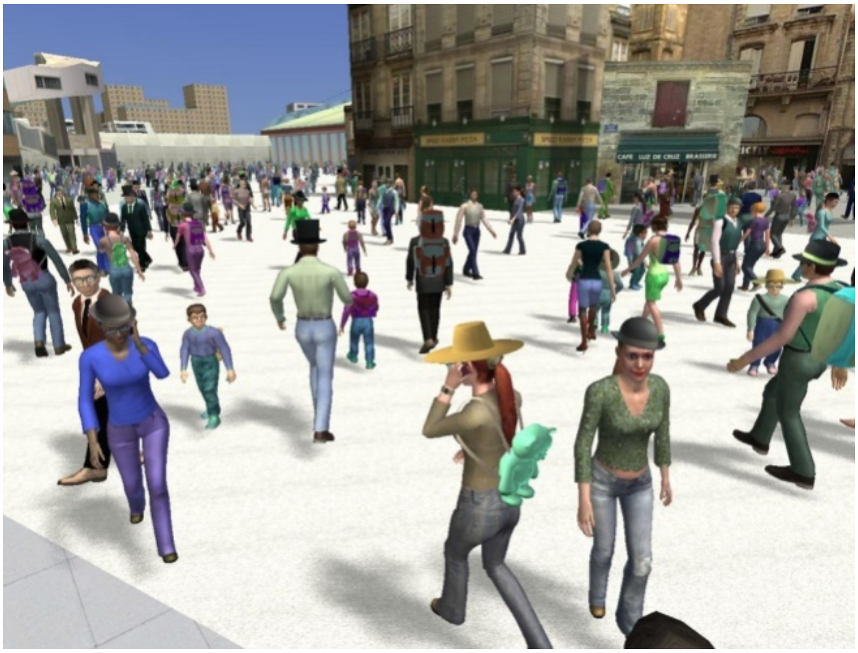


Fig. 5. Crowd simulation

In our Lab, Glardon et al. [19] developed an integrated walking and running engine able to extrapolate data beyond the space described by the PCA basis. Figure 4 shows an example. In this approach, the Principal Component Analysis (PCA) method is used to represent the motion capture data in a new, smaller space. As the first PC's (Principal Components) contain the most variance of the data, an original methodology is used to extract essential parameters of a motion. This method decomposes the PCA in a hierarchical structure of sub-PCA spaces. At each level of the hierarchy, an important parameter (personification, type of motion, speed) of a motion is extracted and a related function is elaborated, allowing not only motion interpolation but also extrapolation. Moreover, effort achieved concerning the possibility not only to qualify a parameter but also to quantify it improves this method in comparison to other similar works. Generic animation, applicable to any kind of human size, is another important aspect of such a research.

The method cannot only normalize a generated motion, it may animate human with a large range of different size (Fig.4). In order to have a complete locomotion engine, transitions between different motions should be handled. IK was also used to prevent feet sliding by exploiting the predictive capability of the model. The method was used as a basis for crowd simulation [20]. Figure 5 shows an example of the use of walking engine for crowds.

5 Low Dimensional Human Motion Synthesis

Carvalho et al. [21] recently proposed a new approach for interactive low dimensional human motion synthesis, by combining motion models and prioritized inverse kinematics. They developed a constrained optimization framework to solve the synthesis problem within a low-dimensional latent space. In this space, a movement enforcing a new set of user specified constraints could be obtained in just one step. They demonstrated the performance, robustness, and simplicity of their approach by synthesizing various styles of golf swings (Fig.6); and by comparing the quality of the synthesized results against a per-frame PIK technique.

Building motion models instead of pose models demonstrated two important advantages, i.e., the system could automatically synthesize the movement as a whole without introducing artifacts and impose continuity between frames through the latent space. Given an appropriate database containing motion samples of the same style, they can build local models and use them to improve motion synthesis. The experiments showed that local models provided faster synthesis results compared to multi-pattern models. Likewise, by giving constraints different priorities also demonstrated an improvement in synthesis performance. This approach is well suited to deal with deformations and retargeting problems. It can also be exploited for a wider range of coordinated motions (e.g., baseball, tennis, jumping, boxing, and others).

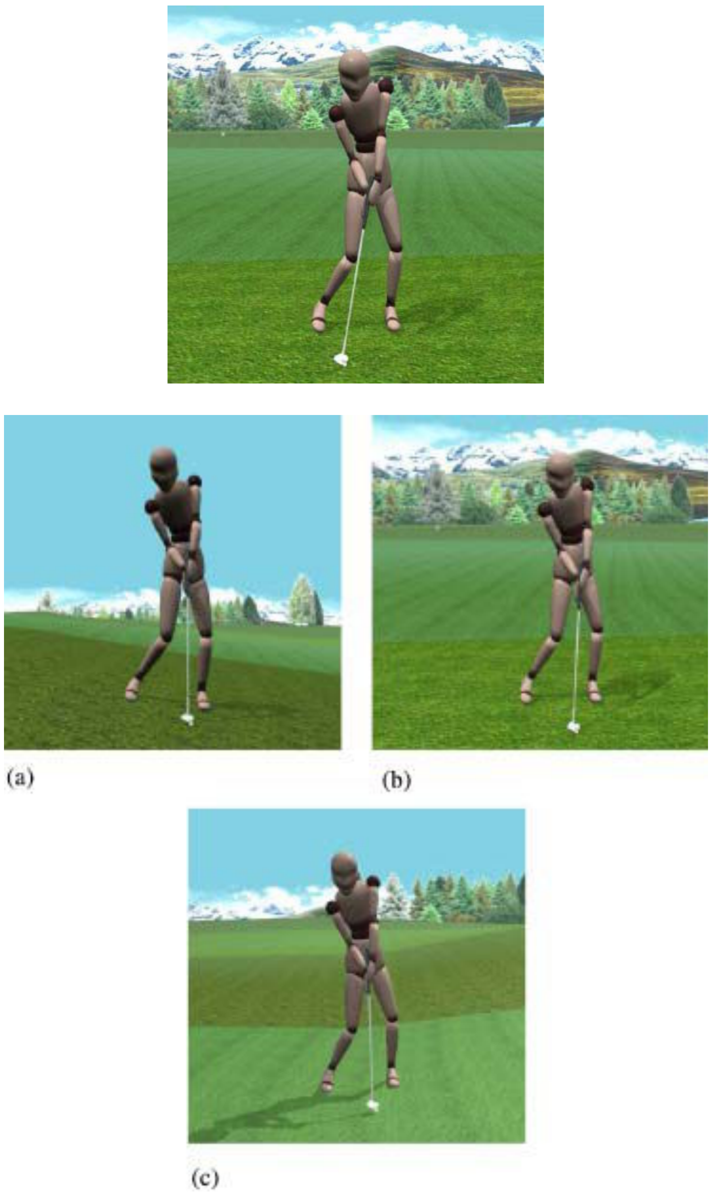


Fig. 6. Synthesized motions, by using our system, from the swing motion shown in the top Figure. (a) 7◦ down slope ground; (b) flat ground; (c) 7◦ up slope ground.

6 Autonomous Behavior

We generally consider autonomy as the quality or state of being self-governing. Perception of the elements in the environment is essential, as it gives the agent the awareness of what is changing around it. It is indeed the most important element that one should simulate before going further. Most common perceptions include (but are not limited to) simulated visual and auditive feedback. Adaptation and intelligence then define how the agent is capable of reasoning about what it perceives, especially when unpredictable events happen. On the other hand, when predictable elements are showing up again, it is necessary to have a memory capability, so that similar behaviour can be selected again. The subtle and spontaneous gestures are part of low-level behaviors, and they are essential to communicate any message and convey emotions. These ideas are just beginning to be explored by the scientific community, some studies focus on non-human characters, like dogs or other virtual animals [22], while others put special attention on human-like facial gestures.

The low-level behavioral animation is an open research avenue. Creating virtual characters able to select the action to perform in an automatic way is not new. State of the art autonomous characters and Virtual Humans are able to perform a variety of tasks, and have some level of autonomy in their decision making, but still they don't look like real people. One of the problems is that the virtual humans don't display the subtle gestures and mannerisms that characterize a real human being: facial expressions, body language, etc.

Our *Artificial Life Environment* [23] framework equips a Virtual Human with the main virtual sensors, based on an original approach inspired by neuroscience in the form of a small *nervous system* with a simplified control architecture to optimise the management of its virtual sensors as well as the virtual perception part. The processes of filtering, selection and simplification are carried out after obtaining the sensorial information; this approach allows us to obtain some persistence in the form of a "cognitive map" and also serves as a pre-learning framework to activate learning methods of low and high level concerning the behaviour of an Virtual Human. The objective pursued is to allow the Virtual Human to explore virtual environments until then unknown and to construct *mental structures and models, cognitive maps or plans from this exploration.* Once its representation has been constructed, this knowledge can be passed on to other Virtual Humans. The Autonomous Virtual Human perceives objects and other Virtual Humans with the help of its virtual environment, which provides the information concerning the nature and position of the latter. The behavioural model to decide which actions the Virtual human should take such as walking, handling an object, etc then uses this information. But, at this stage, how to generate the convenient actions at the right time ? Only true motion models could do it but, unfortunately this is only achieved through motion capture sequences or key-frame based animation. For example, Noser et al. [24] produced vision-based tennis players; they were extremely efficient in terms of strategy and able to know exactly where to be located on the court at any time; but it was impossible to generate the smooth player motion required. For this reason, the players were simplified into rigid racquets. It should be noted that video games offering smooth football or tennis players use thousands of motion capture sequences, a very expensive strategy only possible for the movie and the videogame industry..

7 Conclusions

Character Animation and especially Human Animation is still very dependent on Motion Capture. Movie and Game industry use the databases of pure Motion capture sequences. The only “models” used today are strongly based on Motion Capture; they extensively rely on statistical methods like PCAs and/or use IK-based methods. The pure model is still an academic dream for researchers in Animation. Motion capture will still have a lot of beautiful days in the future.

In order to develop truly interactive multimedia systems with Virtual Humans, games, and interactive movies, we need a flexible way of animating these Virtual Humans. Altering motion obtained from a motion capture system is not the best solution. Only computational models can offer this flexibility unless powerful motion retargeting methods are developed, but in this case they will look similar to computational models.

Acknowledgments. This research has been partly sponsored by the Swiss National Research Foundation. The author would like to thank Dr. Ronan Boulic, Dr. Pascal Glardon, Barbara Yersin, Jonathan Maim, Schubert Carvalho for their work in VRlab.

References

1. Treuille A., Lee Y., Popovic Z., Near-optimal Character Animation with Continuous Control. In: Proc. SIGGRAPH 2007 (2007)
2. Multon, F., France, L., Cani-Gascuel, M.-P., Debunne, G.: Computer animation of human walking: a survey. The Journal of Visualization and Computer Animation (1999)
3. Emering, L., Boulic, R., Thalmann, D.: Interacting with Virtual Humans through Body Actions. IEEE Computer Graphics and Applications 18(1), 8–11 (1998)
4. Herbelin, B., Ponder, M., Thalmann, D.: Building exposure: synergy of interaction and narration through the social channel. Presence 14(2), 234–246 (2005)
5. Garcia-Rojas, A., Vexo, F., Thalmann, D.: Semantic Representation of Individualized Reaction Movements for Virtual Human. International Journal of Virtual Reality 6(1), 25–32 (2007)
6. Baerlocher, P., Boulic, R.: An inverse kinematics architecture enforcing an arbitrary number of strict priority levels. Vis. Comput. 20(6), 402–417 (2004)
7. Boulic, R., Mas, R., Thalmann, D.: A robust approach for the control of the center of mass with inverse kinetics, Computers and Graphics. Mobile Computing 20(5), 693–701 (1996)
8. Wooten, W., Hodgins, J.: Simulating leaping, tumbling, landing and balancing humans. In: Proc. IEEE International Conference on Robotics and Automation, pp. 656–662 (2000)
9. Boulic, R., Ulciny, B., Thalmann, D.: Versatile walk engine. J. Of Game Development, 29–50 (2004)
10. Bruderlin, A., Calvert, T.: Knowledge-driven, interactive animation of human running. In: Proceedings of the Graphics Interface 1996 conference, pp. 213–221 (1996)
11. Sun, H., Metaxas, D.: Automating Gait Generation in SIGGRAPH 2001: Proceedings of the SIGGRAPH 2001 conference, pp. 261–270 (2001)
12. Chung, S., Hahn, J.: Animation of Human Walking in Virtual Environments. In: Proceedings of Computer Animation 1999 conference, pp. 4–15 (1999)
13. Le Callennec, B., Boulic, R.: Interactive motion deformation with prioritized constraints. Graphical Models 2006 68(2), 175–193 (2004); Special Issue on SCA 2004

14. Gleicher, M.: Comparing constraint-based motion editing methods. Graphical models 63(2), 107–134 (2001)
15. Lee, J., Shin, S.Y.: A hierarchical approach to interactive motion editing for human-like figures. In: Proceedings of ACM SIGGRAPH 1999 (1999)
16. Safonova, A., Hodgins, J.K., Pollard, N.: Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. ACM Transactions on Graphics 23(3), 514–521 (2004)
17. Grochow, K., Martin, S.L., Hertzmann, A., Popovic, Z.: Style-based inverse kinematics. ACM Transactions on Graphics 23(3), 522–531 (2004)
18. Shin, H.J., Lee, J.: Motion synthesis and editing in lowdimensional spaces: research articles. Computer Animation and Virtual Worlds 17(3–4), 219–227 (2006)
19. Glardon, P., Boulic, R., Thalmann, D.: Robust on-line adaptive footplant detection and enforcement for locomotion. The Visual Computer 5(6), 194–209 (2006)
20. Pettre, J., De Heras Ciechomski, P., Maim, J., Yersin, B., Laumond, J.P., Thalmann, D.: Real-time navigating crowds: Scalable simulation and rendering. Computer Animation and Virtual Worlds 17(3-4), 445–455 (2006)
21. Carvalho, S., Boulic, R., Thalmann, D.: Interactive Low-Dimensional Human Motion Synthesis by Combining Motion Models and PIK. Computer Animation & Virtual Worlds 18 (2007)
22. Isla, D., Blumberg, B.: Object Persistence for Synthetic Creatures. In: the Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), Bologna, Italy (July 2002)
23. Conde, T., Thalmann, D.: An integrated perception for autonomous virtual agents: active and predictive perception. Computer Animation and Virtual Worlds 17(3-4), 457–468 (2006)
24. Noser, H., Thalmann, D.: Sensor-based synthetic actors in a tennis game simulation. Visual Computer 14(4), 193–205 (1998)

Informed Use of Motion Synthesis Methods

Herwin van Welbergen[1], Zsófia Ruttkay[1], and Balázs Varga[2]

[1] HMI, Dept. of CS, University of Twente, Enschede, The Netherlands
[2] Pázmány Péter Catholic University, Budapest
{welberge,zsofi}@ewi.utwente.nl

Abstract. In virtual human (VH) applications, and in particular, games, motions with different functions are to be synthesized, such as communicative and manipulative hand gestures, locomotion, expression of emotions or identity of the character. In the bodily behavior, the primary motions define the function, while the more subtle secondary motions contribute to the realism and variability. From a technological point of view, there are different methods at our disposal for motion synthesis: motion capture and retargeting, procedural kinematic animation, force-driven dynamical simulation, or the application of Perlin noise. Which method to use for generating primary and secondary motions, and how to gather the information needed to define them? In this paper we elaborate on informed usage, in its two meanings. First we discuss, based on our own ongoing work, how motion capture data can be used to identify joints involved in primary and secondary motions, and to provide basis for the specification of essential parameters for motion synthesis methods used to synthesize primary and secondary motion. Then we explore the possibility of using different methods for primary and secondary motion in parallel in such a way, that one methods informs the other. We introduce our mixed usage of kinematic an dynamic control of different body parts to animate a character in real-time. Finally we discuss motion Turing test as a methodology for evaluation of mixed motion paradigms.

1 Introduction

There are different kinds of human motions, considering the goals and meanings involved, such as communicative and manipulative hand gestures or locomotion. These are all needed in virtual character enhanced applications, like a game. When synthesizing such motions, there are requirements beyond the function: the motion should be realistic, life-like. Motion with the same function repeated by the same person should differ in small motion details.

When observing the entire body, one may notice big motions (like the waving hand), and small motions like balancing with the torso. In this paper we refer to the big motion which is characteristic of the goals and meanings involved as *primary motion*, and the additional details as *secondary motion*. Roughly speaking, when synthesizing the primary motion only, the virtual characters motion can be understood. But without the secondary motions, it will look robot-like and unnatural.

© Springer-Verlag Berlin Heidelberg 2008

From a technological point of view, there are different methods at our disposal: motion capture and retargeting, key-framed or procedural kinematic animation, force-driven dynamical simulation, or the application of Perlin noise. Which method to use, and how to gather the information needed to specify the parameters for the individual methods?

In practice, often the technological considerations – typically, production tools and resources available and the required processing speed – are the factors to justify a single solution. Moreover, as each of the methods are cumbersome, researchers basically concentrate on one of the approaches and improving its algorithms. This is characteristic of the rather disjoint work going on in the 'mocap world' and the 'motion generation' world. The dichotomy is also to be noticed when looking at the application domains. In medical applications one can find high quality dynamical simulation to study walking under different conditions, in movies the quality of (virtual human) motion is high due to using mocap, while in the conversational agents domain, procedural models are used.

The main topic of our paper is to investigate how different techniques can be used to synthesize both primary and secondary motions. Using combinations of these techniques, we aim to come close to the illusion of full realism of the behavior, while also being responsive to the environment the motion is executed in and reactive to the gamer's behavior in real time. We consider informed usage of motion synthesis methods, in two interpretations of 'informedness'. First, one can use motion capture data to inform the (models for) generative methods, particularly, to decide about the primary and secondary motions. Second, one can use multiple methods together for different body segments for primary and secondary motions in such a way that one method informs other(s), see Fig. 1.

We apply our motion synthesis algorithms on our virtual conductor [1] and a reactive virtual trainer [2]. The virtual conductor can conduct human musicians in a live performance interactively. Using knowledge of the musical piece (tempo, volume, the different voices, etc...) he leads the musicians through the piece and corrects them when certain types of mistakes occur. The Reactive Virtual Trainer is capable of presenting physical exercises that are to be performed by a human, monitoring the user and providing feedback at different levels. Both applications require real-time adaptation and precise control of the timing of motion. The primary motion of the conductor consists of conducting gestures executed with one or both arms and/or the head. The secondary motion of the conductor is modeled as balancing motion on the lower body. For the trainer, the primary and secondary motions are exercise depended.

In the next section, we introduce each motion synthesis method shortly, and give an overview of state of the art of using multiple methods in one framework. Then, in section 3, we discuss how motion capture can be used to identify joints to be modeled by different methods, and how to tune the parameters of these methods. In section 4, we discuss the potentials of mixing multiple methods to animate different parts of the body. We demonstrate the idea by an example from our ongoing work, showing mixed usage of kinematic an dynamic control of different body parts to animate a character in real-time. The paper finishes

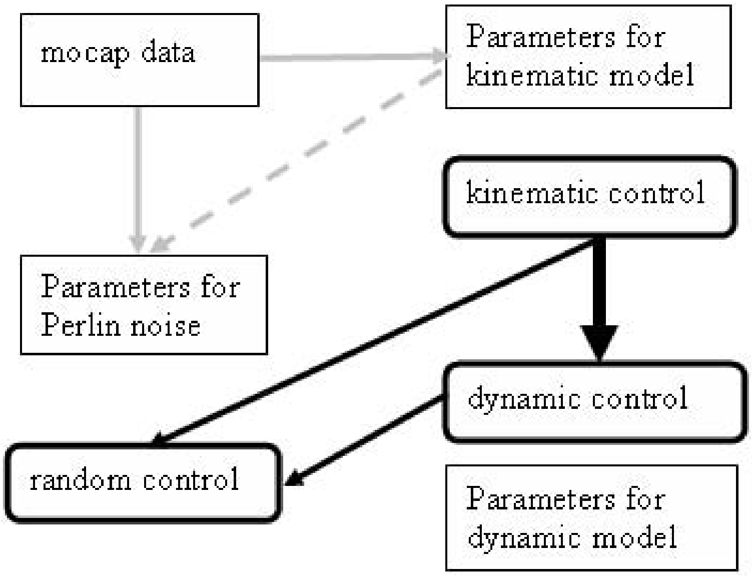

mocap data
Parameters for kinematic model
kinematic control
Parameters for Perlin noise
dynamic control
random control
Parameters for dynamic model

Fig. 1. The three procedural control paradigms, gray arrows showing off-line parameter acquisition based on information from other methods. The other arrows show the possibility for information exchange for real-time parallel tuned usage of multiple methods. The thick arrow indicates the novel method discussed we implemented.

by discussing the motion Turing test, as the evaluation paradigm we plan to use in the future to evaluate mixed motion models.

2 Related Work

Current computer animation techniques make use of kinematic and/or dynamic techniques for motion generation.

Motion editing uses recorded motion as a basis and creates variations on this motion to adapt it to specific needs. This kinematic technique employs the detail of captured motion but only allows very small adaptations. It is relatively simple to use slight motion adaptations to adhere to strict time constraints [3]. Typically, the actor that is motion captured does not have the same limb size as the VH that is to be animated. Therefore, it is necessary to retarget the motion of the actor to make it work on this VH [4]. Sensor noise, motion retargetting and motion adaptation causes physical anomalies on edited motion. One of such anomalies is footskate: the VH's foot slides on the floor after the VH plants it, rather than remaining tightly in place [5]. Another typical artifact of motion editing is the violation of balance, which results in a VH that not seems to be situated in the virtual world but rather seems glued in front of it [6].

Procedural animation is a kinematic technique that uses mathematical formulas that prescribe the change of posture over time, given a set of movement parameters values. Procedural animation can be used to directly control the rotation of joints [7]. A typical application is at a slightly higher level, for generating gestural behavior: the movement path of hands through space is defined mathematically, and the appropriate arm joints are computed automatically [8, 9, 10]. This approach is very adaptable: it offers precise timing and real-time parameterization using a large number of parameters. However, it is hard to incorporate movement details such as those found in motion recordings into the mathematical formulas that steer procedural motion. Physical believability has to be authored explicitly in the formulas. Typically, only a subspace of parameter values yields physically realistic motion.

In *dynamic simulation*, the VH is controlled by applying torques on the joints and a force on the root. A physical simulator then moves the VH's body using Newtonian dynamics, taking friction, gravity and collisions into account. The process of finding accelerations of rigid bodies, based on forces and torques is called *forward dynamics*. A dynamic controller can provide control in real time [11], if the dynamical model of the human is kept simple. Such a controller calculates the torques that will move the VH toward a desired state, based on a heuristic movement model. The input to such a controller is the desired value of the VH's state, for example desired joint rotations or the desired position of the VH's center of mass (CoM). The goal of the system is to minimize the discrepancy between the actual and desired state. The controller can deal with perturbations. For example, if balance is disturbed by a push, the controller automaticly guides the VH back to a balanced pose. However, in general, the controller can not predict when the desired state is reached, or if it will be reached at all. Motion generated by simulation lacks the kind of detail that is seen on captured motion. While the motion is physically correct, this alone is often not enough for movements to be human-like. Therefore, dynamic simulation is mainly used to generate human motion that is physically constrained, and in which interaction with the environment is important, such as motion by athletes [11], stunt men [12], or people falling over [13].

2.1 Combining Dynamic and Kinematic Animation

The previously introduced animation paradigms each have their own advantages, so it would be beneficial to combine them in a single motion system. Existing hybrid systems [12, 13] typically switch between motion editing and dynamic simulation, depending on the current situation's needs.

However, in many situations, the different positive features of the generation paradigms are needed at the same time, but at different body parts. For example, while standing and gesturing, our lower body is in contact with the ground and has to support the upper body, showing a physically realistic balanced pose which would be nice to model with dynamic simulation, while arm gestures require motion detail and tight time synchronization to speech, which would be best achieved by kinematic motion.

In [14] it is shown how inverse and forward dynamics can be combined in a single dynamic animation system, assuming that either the joint acceleration or the joint torque is known for each joint, at each time frame. A similar approach is commonly used in biomechanics to visualize the biomechanical movement model of interest on some body parts (using joint torques), enhanced with known motion on other body parts (using kinematic motion) [15]. Our work on mixed kinematics and dynamics revives the ideas in [14] and shows how they can be applied in an efficient real-time motion system.

2.2 Tracking Kinematic Motion

Motion tracking [16], uses a controller to compute the torque on each joint. The desired state for this controller is the desired rotation of the joint, as specified in by the kinematic data (e.g. mocap). Motion capture noise, retargetting errors, tracking errors and environmental changes can easily disturb the balance of a character whose full body is animated using a tracking controller. Therefore an explicit dynamic balancing model is needed. Because tracking makes use of dynamic controllers, the motion generated by these methods has a time-lag with that specified in the motion capture data. Tracking controllers are computationally more demanding than our mixed kinematic/dynamic method, but provide collision with the environment for the tracked body parts.

3 Modeling Informed by Motion Capture

In the recent years, we have been using motion capture technology to analyze different aspects of hand gestures and full body physical exercise motions [17,18]. We use the results to inform the different modeling paradigms used to synthesize motions. The domain of our investigations are physical exercises for a Reactive Virtual Trainer application [2] where physical realism is of major importance, repetitive motions like clapping [17] or conducting [1], and some communicative gestures like hand-shaking [19]. In all cases, we recorded several samples of the same gesture, by the same subject.

3.1 Identification of Primary and Secondary Joints

The primary and secondary joint motions were identified by looking at the standard deviation of the coordinates of marker positions for each joint, normalized by maximum joint extension. For each motion, the joints are classified as primary, secondary or unused in an automatic way. Currently, the two separating thresholds are to be given manually as a first step, but fully automatic clustering may be used in the future. For all but one recording of rhythmic physical exercise samples, this method identifies the 1-6 primary joints faithfully. That is, the judgment of the classification system corresponds to the judgment of an expert.

3.2 Kinematic Model Based on Motion Capture

In ongoing work, we wish to use the captured time functions of the motion of primary joints to detect certain kinematic characteristics of the motion, and to define parameters for kinematic models. As of motion phases of a hand gesture or exercise, we adopt the terminology used for communicative gestures [20], identifying *preparation*, *pre-stroke hold*, *stroke*, *post-stroke hold* and *retraction*. As an exercise is to be performed in a repetitive way, the retraction is identical with the preparation of the next exercise. By analyzing the captured data, we are to identify:

1. tempo
2. timing of the preparation, hold and stroke phases
3. amplitude, expressed by the maxim values of the displacement of certain body part
4. synchrony

By having recorded tempo variants of the same motion, we also get information of extreme and default tempi, and the correlation between tempo and amplitude, and tempo and timing of stages. This quantitative information is to serve as a basis for kinematic models. The timing information of the phases is gained by identifying points in the motion on which the speed crosses a predefined minimum threshold (see Fig 2).

Synchrony is to identify if (a subsets of) the joints involved move in sync. This is to be decided by analyzing the stroke times of the different joints. In case of physical exercises, a few synchrony patterns are common [2].

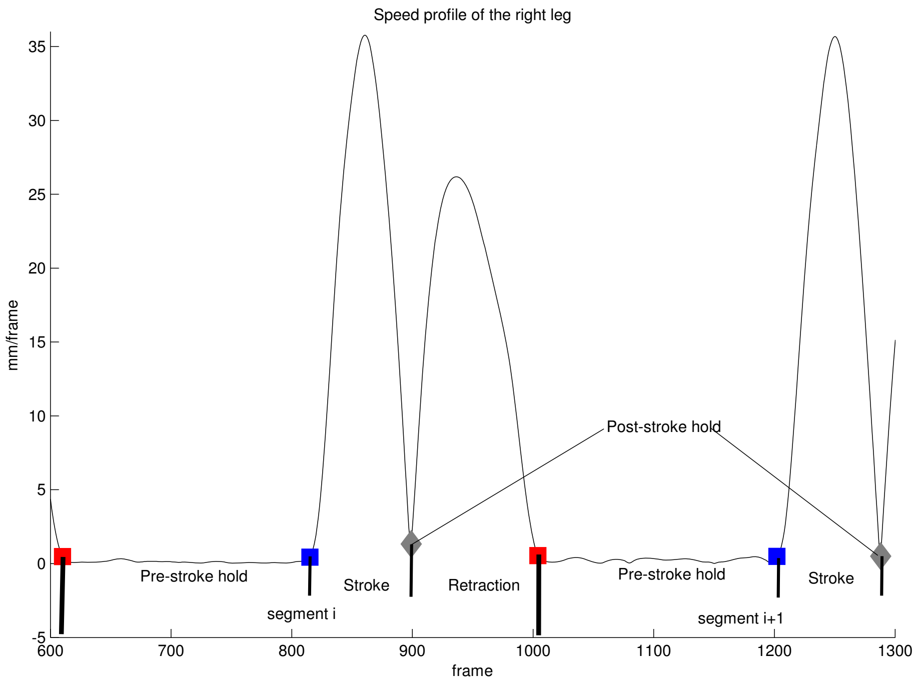


Fig. 2. Motion phases are detected from the speed profile

The physical and procedural models themselves are typically created on the basis of models from biomechanics or behavior science, rather than directly basing them off motion capture. Motion capture information is used to identify (among others) timing, amplitude and synchrony parameters and to select a parameterized model that fits the observed motion. The parameters that steer such a model are designed to be intuitive for motion authors, but are often related. Motion capture can serve as a way to find dependencies between these parameters. For example, we have shown that the movement path of the hand decrease linearly with the tempo in a clapping task [17]. A change of one parameter value then changes all parameters values that are related to it. If a motion generation process specifies the value of more than one parameter, conflicts might arise. These conflicts can be solved in several ways, for example by finding some kind of 'best fit' of parameters values, weighted by their importance.

3.3 Analyzing Variability and Symmetry

We gathered qualitative and quantitative information on the variation between repeated motions with the same function. Even in case of physical exercises performed by expert, there is a small variation in performance. Similar considerations hold for symmetry (see Fig. 3). The amount of variation can depend movement parameters. For example, Fitts' law [21] states that quick pointing movements are neceserely less precise than slower pointing movements.

Small, but consistent phase differences can occur in symmetrical movement. For example, we have shown that right-handed subjects 'lead' a clapping movement with their right hand [17]. The mean phase difference and variation of phase difference can vary with movement parameters. For example, we found that the standard deviation of the phase difference between hands in a clapping task increases with tempo [17].

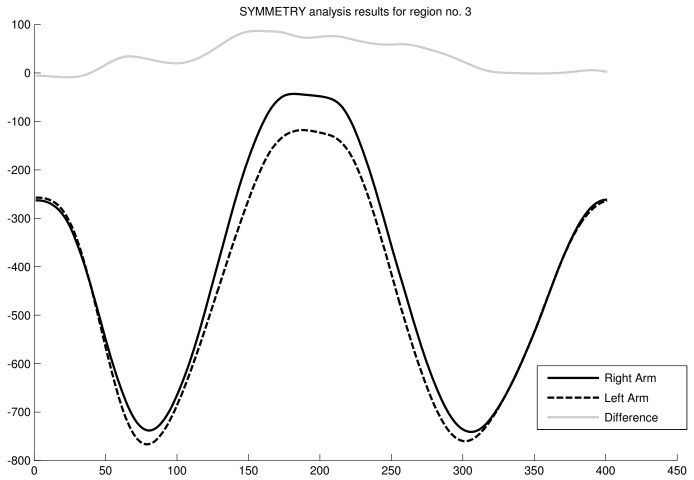


Fig. 3. Symmetry analysis of an exercise motion

4 Using Dynamic Simulation and Kinematics in Parallel

In [22], we show how to simultaneously combine dynamic motion with kinematicaly driven motion (like motion capture, procedural motion). In case of the conductor, the body is segmented in one physically steered group of joints P, located at the lower body and three chains of kinematicly controlled joints, $K_{rightarm}$, $K_{leftarm}$ and K_{head} rooted at the right shoulder, left shoulder and neck respectively (see Fig. 4). The movement of the lower body is steered by the dynamic balance controller, described in [11].

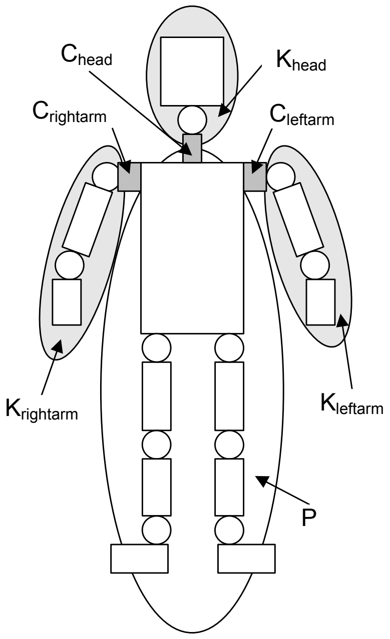


Fig. 4. Kinematically and dynamically steered joints in the conductor

The torque the kinematic chains exert at connectors ($C_{rightarm}$, $C_{leftarm}$ and C_{head}) is calculated using inverse dynamics. The reactive torques are then applied to the lower body. To calculate this torques, we need to know the velocity and acceleration of the connectors. However, the velocity and acceleration of a connector is dependent on the movement of all joints in the body, and can only accurately be calculated by an algorithm that takes the accelerations of all joints in K and the torques of all joints in P into account simultaneously. Rather than directly calculating the acceleration of the connectors at the current frame, we approximate torques that each K exerts on P using the acceleration and velocity of connector at the *previous* frame. This results in a slightly more efficient algorithm than first proposed in [14], both in terms of calculation time and memory bandwidth. More importantly, it allows us to make use of any existing real-time forward dynamics engine for mixed motion generation.

Fig. 5 shows the combination of our physical balance model with a procedurally generated large arm swing. The results of our system are subtle and therefore hard to capture on a series of images. We refer the interested viewer to the demonstration videos [1] to see our system in action with a combination of motion captured arm movements or procedural conducting gestures and the balance model.

5 Further Work

5.1 Generating Variability

In some of our applications (conductor, fitness trainer) the same movement is to be repeated many times. If such movement looks exactly the same for repetition, the believability of our VH is destroyed. We intend to use one of the existing approaches from the literature [7,23,24] to model motion variability. Our motion capture analysis provides us with information on the amount of variability in each motion, on what joint or motion parameter this variability is to be applied and how parameter values affect its size.

5.2 Timing of Motion Phases

We have timing information on each movement phase in our exercise motion. We intend to explore the relation of the duration of each phase length with other parameters, such as movement tempo or amplitude. In our preliminary work on the analysis of clapping [17], we found an invariance in the relative timing of the different phases of clapping movement: no matter the tempo, the same percentage of time was spend in each movement phase. These percentages were slightly different for different subjects.

5.3 Evaluating Movement Models

VHs usually do not have a photo-realistic embodiment. Therefore, if the naturalness of VH animation is evaluated by directly comparing moving humans with a moving VH, the embodiment could bias the judgment. To remedy this, motion captured human movement can be casted onto the VH, thus showing the different motions on identical virtual body. This motion is then compared with generated animation. Typically this is done in an informal way. A *motion Turing Test* [25] could be used to do this more formally. In such a test, subjects are shown generated movement and similar motion captured movement, displayed on the same VH. Then they are asked to judge whether this was a 'machine' moving or a real human.

However, such a direct human judgment is not sufficient to measure the naturalness of motion. Even if a certain movement is judged as natural, an unnatural artifact that is not noticed consciously can still have a social impact [26]. Characters with an unnatural motion can be evaluated as less interesting, less pleasant,

[1] Available from: http://www.herwinvanwelbergen.nl/phd/mixed/mixed.html

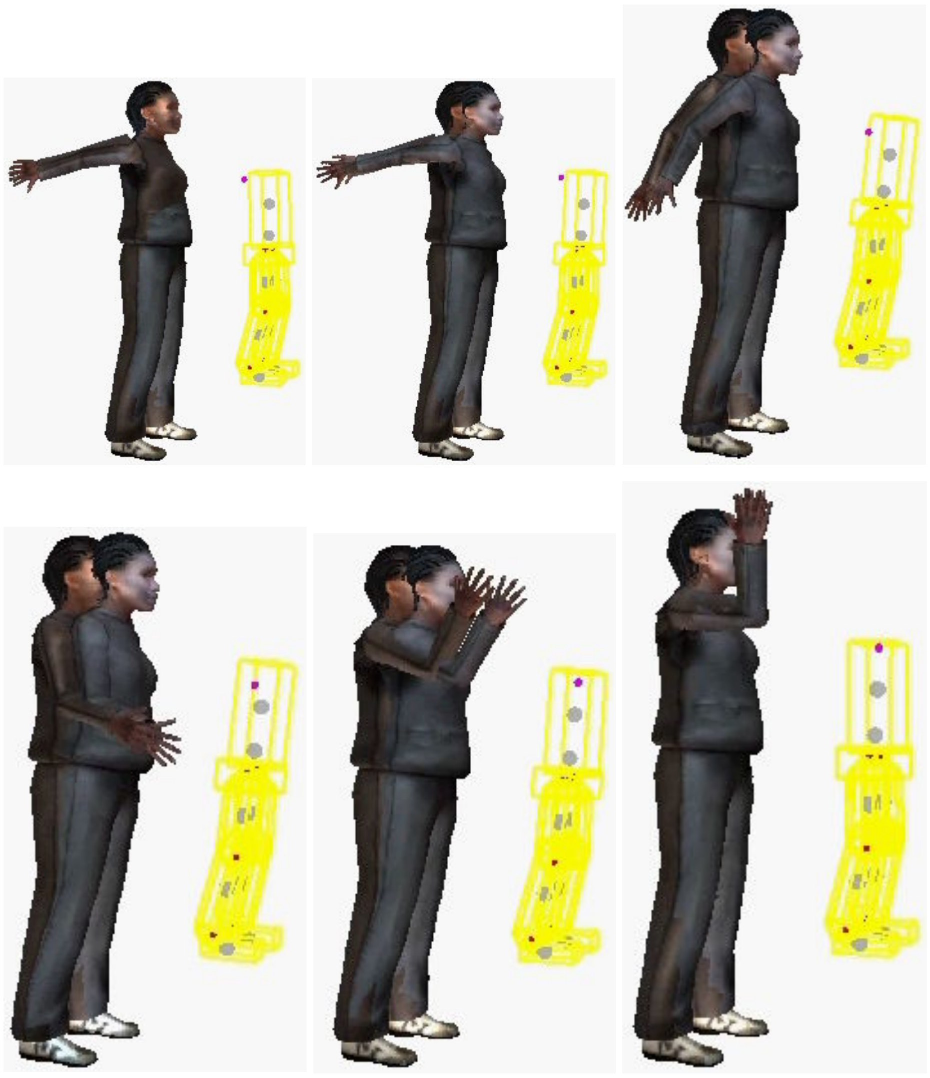


Fig. 5. Mixing a kinematic arm swing with dynamic balancing. The right side of each picture shows the visualization of the physical model of the lower body of the VH. The left side shows a VH with physical lower body control, overlayed with one that does not move the lower body. The counter-clockwise angular acceleration of the arm in the upper three frames causes a clockwise torque on the trunk, which makes the upper body bend forward. The clockwise angular acceleration of the arm on the lower three frames causes the body to bend backward.

less influential, more agitated and less successful in their delivery. So, while a VH Turing test is a good first indication of naturalness (at least it looked human-like), further evaluation should determine if certain intended aspects of the motion are delivered. Such aspects could include showing emotion, enhancement of the clearness of a spoken message using gesture, showing personality, etc.

Our simultaneous motion mixing techniques allow us to let a motion model steer a part of the body, and let a motion capture recording steer the remaining parts. We can then use the motion Turing test to compare the motion generated by the movement model combined with motion capture with the same motion generated solely by motion capture. In a similar way, we can test if a certain aspect of motion is important for naturalness, by using a model that either removes this aspect, or replaces it by noise. In a later stage, we plan to use this approach to test combinations of motion models that were evaluated to work well in isolation.

Further on, we would like to explore the parameter space that yields natural looking procedural or dynamical motion. If the parameter set is small, this natural parameter space can be identified by having subjects directly set and evaluate the possible parameter values [23].

Acknowledgments

This research has been supported by the GATE project, funded by the Netherlands Organization for Scientific Research (NWO) and the Netherlands ICT Research and Innovation Authority (ICT Regie). The motion capture recording and analysis was carried out in the framework of a Short Mission supported by the EU Action COST2102.

References

1. Reidsma, D., Nijholt, A., Bos, P.: Temporal interaction between an artificial orchestra conductor and human musicians. ACM Computers in Entertainment (to appear, 2008)
2. Ruttkay, Z., van Welbergen, H.: Elbows higher! performing, observing and correcting exercises by a virtual trainer. In: Proceedings of Intelligent Virtual Agents, Tokyo, Japan (to appear) (September 2008)
3. Witkin, A., Popovic, Z.: Motion warping. In: SIGGRAPH, pp. 105–108. ACM Press, New York (1995)
4. Gleicher, M.: Retargetting motion to new characters. In: SIGGRAPH, pp. 33–42. ACM Press, New York (1998)
5. Ikemoto, L., Arikan, O., Forsyth, D.A.: Knowing when to put your foot down. In: Proceedings of Interactive 3D graphics and games, pp. 49–53. ACM Press, New York (2006)
6. Tak, S., Song, O.-Y., Ko, H.-S.: Motion balance filtering. Computer Graphics Forum 19(3) (2000)
7. Perlin, K.: Real time responsive animation with personality. IEEE Transactions on Visualization and Computer Graphics 1(1), 5–15 (1995)
8. Chi, D.M., Costa, M., Zhao, L., Badler, N.I.: The EMOTE model for effort and shape. In: SIGGRAPH, pp. 173–182. ACM Press/Addison-Wesley Publishing Co., New York (2000)

9. Neff, M., Kipp, M., Albrecht, I., Seidel, H.P.: Gesture modeling and animation based on a probalistic recreation of speaker style. Transactions on Graphics (to appear, 2008)
10. Hartmann, B., Mancini, M., Pelachaud, C.: Implementing expressive gesture synthesis for embodied conversational agents. In: Gibet, S., Courty, N., Kamp, J.-F. (eds.) GW 2005. LNCS, vol. 3881, pp. 188–199. Springer, Heidelberg (2006)
11. Wooten, W.L., Hodgins, J.K.: Simulating leaping, tumbling, landing, and balancing humans. In: ICRA, pp. 656–662. IEEE, Los Alamitos (2000)
12. Faloutsos, P., van de Panne, M., Terzopoulos, D.: The virtual stuntman: dynamic characters with a repertoire of autonomous motor skills. Computers & Graphics 25, 933–953 (2001)
13. Zordan, V.B., Macchietto, A., Medina, J., Soriano, M., Wu, C.C.: Interactive dynamic response for games. In: Proceedings of the SIGGRAPH symposium on Video games, pp. 9–14. ACM Press, New York (2007)
14. Isaacs, P.M., Cohen, M.F.: Controlling dynamic simulation with kinematic constraints. In: SIGGRAPH, pp. 215–224. ACM Press, New York (1987)
15. Otten, E.: Inverse and forward dynamics: models of multi-body systems. Philosophical Transactions of the Royal Society 358(1437), 1493–1500 (2003)
16. Zordan, V.B., Hodgins, J.K.: Motion capture-driven simulations that hit and react. In: Proceedings of the Symposium on Computer Animation, pp. 89–96. ACM Press, New York (2002)
17. van Welbergen, H., Ruttkay, Z.: On the parameterization of clapping. In: Gesture Workshop, Lisbon, Portugal (July 2007)
18. Varga, B.: Movement modeling and control of the reactive virtual trainer. Master's thesis, Peter Pazmany Catholic University, Dept. of Information and Technology, Budapest, Hungary (2008)
19. Ruttkay, Z., van Welbergen, H.: Let's shake hands! on the coordination of gestures of humanoids. In: Artificial and Ambient Intelligence, Newcastle University, Newcastle upon Tyne, UK, pp. 164–168 (April 2007)
20. McNeill, D.: Hand and Mind: What Gestures Reveal about Thought. University of Chicago Press, Chicago (1995)
21. Fitts, P.M.: The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology 47(6), 381–391 (1954)
22. van Welbergen, H., Zwiers, J., Ruttkay, Z.: Real-time animation using a mix of dynamics and kinematics. In: Symposium On Computer Animation (submitted, 2008)
23. Bodenheimer, B., Shleyfman, A.V., Hodgins, J.K.: The effects of noise on the perception of animated human running. In: Computer Animation and Simulation (September 1999)
24. Egges, A., Molet, T., Magnenat-Thalmann, N.: Personalised real-time idle motion synthesis. In: Pacific Graphics, pp. 121–130. IEEE Computer Society, Washington (2004)
25. Hodgins, J.K., Wooten, W.L., Brogan, D.C., O'Brien, J.F.: Animating human athletics. In: SIGGRAPH, pp. 71–78. ACM Press, New York (1995)
26. Reeves, B., Nass, C.: The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places. Cambridge University Press, New York (1996)

Automatic Estimation of Skeletal Motion from Optical Motion Capture Data

Zhidong Xiao, Hammadi Nait-Charif, and Jian J. Zhang*

National Centre for Computer Animation,
Bournemouth University, United Kingdom
{zxiao,hncharif,jzhang}@bournemouth.ac.uk

Abstract. Utilization of motion capture techniques is becoming more popular in the pipeline of articulated character animation. Based upon captured motion data, defining accurate joint positions and joint orientations for the movement of a hierarchical human-like character without using a pre-defined skeleton is still a potential concern for motion capture studios. In this paper, we present a method for automatically estimating and determining the topology of hierarchical human skeleton from optical motion capture data based on the human biomechanical information. Through the use of a novel per-frame based recursive method with joint angle minimization, human skeleton mapping from optical marker and joint angle rotations are achieved in real time. The output of motion data from a hierarchical skeleton can be applied for further character motion editing and retargeting.

Keywords: Motion capture, Motion tracking, Character mapping, Character animation.

1 Introduction

Motion capture is a technique and a process that digitally record the movements of a live "performer", such as human motion or the movement from an animal in a specified 3D environment. Then the recorded motion data is replicated to animate a computer-generated character. Since motion capture techniques are the most effective means to achieve realistic and convincing actions from a "performer", it has been widely used in computer games, movies where human-like character animation is heavily involved, and other areas where motion analysis is performed.

In animation, the motion of an articulated skeleton character is defined by a world coordinate transformation at the root joint and other relative joints rotational angles in its hierarchy. However, optical motion capture systems only record a set of three dimensional Cartesian points that describe the trajectories of markers attached on the live performer. The recorded three-dimension motion data cannot be used for manipulating a character directly because the output

* Corresponding author.

© Springer-Verlag Berlin Heidelberg 2008

format of motion data does not explicitly construct a hierarchical skeleton. It is difficult to construct a proper skeleton based upon the captured optical motion data, as the transfer of the three dimensional position data to a skeleton joint rotational data is not straightforward. The conversion of the markers Cartesian position data into a hierarchical skeleton movement in the joint space is an important procedure for dealing with motion capture data. After the mapping of the markers' movements to an articulated skeleton's joints rotation, this data conversion process simplifies and formats the motion of the "performer" that is editable and controllable. Such a procedure makes further motion data editing more convenient and shortens the production pipeline of computer animation.

Autodesk Motionbuilder [2], a popular commercial software package, uses a pre-defined skeleton for mapping the optical motion capture data to a hierarchical character motion. During the calibration procedure, the motion capture producer manually adjusts the prototype of the pre-defined character and groups the locations of the markers for data mapping, which is a tedious procedure. The quality of output motion data is based on the experience of the producer. In this paper, we describe a method that will automatically group marker sets to find accurate joint locations in the hierarchy. Since we use a signal-based active motion capture system, there are fewer markers attached on a performer than a passive optical system. In order to find accurate joint positions and obtain more precise joint orientations, our method highly relies on the biomechanical information from human motion [8][16]. Although inverse kinematics techniques [3][17] are good to configure intermediate joints according to the end joint position, they suffer from singularity and initial starting point problems for optimization. Similar to the global technique presented in [15], we present a recursive method with joint angle minimization to calculate the joint rotational angles in the hierarchy.

2 Previous Work

Motion capture systems are able to record realistic and detailed human movement. Numerous researchers focus on the issues related to motion data post-processing techniques. For instance, motion data mapping, motion editing, motion retargeting and motion blending.

In computer animation, animators usually use existing human motion data to drive their own personal designed virtual characters, such as a monster that has a different skeleton prototype or/and different skeleton length from human performers. In order to map human motion data to a custom designed virtual character, some researchers focus on motion retargeting to solve the prototype problem [5][9]. Recently, some researchers in computer graphics are capable of abstracting the characteristic motion, a basis motion data, from an amount of motion database [4][10][14]. They focus on modifying and editing motion data at the behaviour level through statistics based method. Several other researchers took physics into account to modify motion capture data and synthesize convincing and realistic character movement [1][12][13]. These efforts aimed at finding physically convincing character movement in terms of required additional criteria. A motion blending technique was presented to create a long, continuous motion from multi motion clips [7].

The techniques described above are based upon the fact that the skeleton motion are already in a hierarchical joint space. To manipulate hierarchical skeleton for further motion editing, it is important and necessary to obtain joint rotation angles in a hierarchical skeleton from recorded motion capture data. Several researchers have presented their solutions for this specific problem. Bodenheimer et al described a procedure for transforming magnetic motion capture data into an articulated human skeleton movement in which manual measurements of body segments is required [3]. O'Brien et al also estimated the skeleton from a magnetic motion capture system in which each marker has both position and orientation information [11]. By comparison, the data captured from optical motion capture system only record the position trajectory for each marker. The optical motion data are non-rotational, three-dimension Cartesian positions. To construct an articulated skeleton from a camera-based multi-marker optical motion capture system, Silaghi et al presented a method in which the joint position is inferred based upon the adjacent joint link information [15]. Kirk and his colleagues group the markers in different clusters according to the assignment of the markers to body segments, then they use a fitting method to estimate the joint position [6].

Since the captured marker's trajectory describes the movement of physical body parts of a realistic performer, Zordan et al assume that all the markers attached on an actor are driven by force separately. In order to test their assumption, they mapped the marker data into a fixed human character which includes a physically based forward dynamic model. The joint angle information of a fixed limb-length skeleton is obtained from the simulation of the equilibrium state [18].

The optical motion data mapping approaches described above mainly infer the joint position based on the multi-marker information attached at the adjacent body segments. In our method, we define a skeleton prototype automatically. In addition, we can estimate joint angles in real time.

3 Motion Capture and Skeleton Fitting

Currently, passive and active optical motion capture systems are two mainstream system for human-like characters. While both use markers attached to the performer; in a passive system each marker has to be identified continuously during a capture session in order to obtain the location of each marker, whereas in an active system the markers are individually identified through Pulsed-LED (light emitting diodes).

Compared with the tracking techniques used in camera based motion capture system, Ascension ReActor2 is a signal processing based active optical motion capture system. This system has fewer occlusion problems than the passive systems as the markers are instantly recognized after occlusion. This system use only 28 infrared markers for tracking a live performance. Each of the markers is

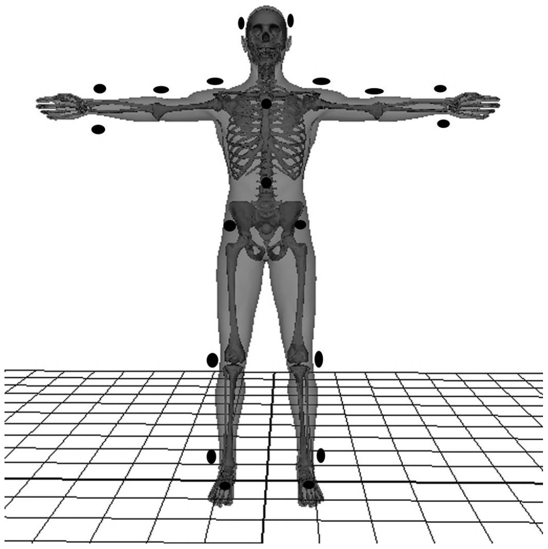


Fig. 1. Topology of human character and some marker positions shown with black dot

attached at the related joint position based on the skeleton hierarchy of a human character. At the calibration, the detectors will be placed at specified positions in 3D space. Detectors will receive light flashed from each marker and determine the location of the markers. The recorded marker positions are discrete three-dimension position points in the time domain. The current capture space is limited to 3×3×3 cubic meters.

For an ideal motion capture, we assume that the markers attached on the body stay relatively static during the motion capture session. The marker location represents the best estimation of the joint positions according to the anatomical topology of a human performer. To avoid manually adjust pre-defined rigid body skeleton for optical marker mapping, in our method, we automatically estimate joint positions from biomechanical information [8]. Figure 1 is the frontal view of a human character model with a skeleton beneath the skin and some marker locations for motion capture arrangement. Our skeleton identification procedure includes two stages: a) identify each marker and group markers for individual rigid body parts, then calculate the actual rotation centre based upon the anatomy knowledge of human body; b) calculate joint rotation angles according to the fitted hierarchical skeleton.

3.1 Marker Identification and Joint Determination

Unlike a passive system, each marker in an active system is able to "communicate" with tracking devices, e.g., signal detectors. The communication is achieved by means of LED pulse. The LED marker signal may be affected more or less by other environment lighting sources during motion capture. As the noise can affect the quality of motion capture, reduce or eliminate the noise effect is important for obtaining a precise marker position. There are wide ranges of frequencies in the electromagnetic spectrum. For infrared, the spectrum range covers from 10^{11} to 10^{14}Hz. Since the frequency spectrum of infrared is known, the removal of the noise can be solved by means of bandpass digital filtering.

To construct a hierarchical human skeleton, an essential task is to find a proper joint position that connects adjacent body segments. Before motion capture, the motion capture producer usually attaches each marker on the performer in a position that is regarded as a reference of the performer's joint. After the detector has received the LED marker pulse, the precise marker position that describes the performer's movement in three-dimensional space is recorded. In an ideal motion capture session, the distance between every two markers that are attached on the same rigid body part would remain the same all the time. This means that the two adjacent markers would define one rigid body link in the hierarchical skeleton. From the biomechanical information of human motion [8][16], the actual human joint location beneath skin that is related to its referential marker group should maintain a constant distance during the movement all the time. To guarantee the joint location is constant for motion data mapping, our approach is similar to that in Kirk et al [6]. We define and minimize a joint penalty function to determine the accurate joint position that is related to its marker group (Equation 1).

$$|(\| {P_i}^k - {P_r}^k \| - d)| \leq \varepsilon \tag{1}$$

In Figure 2, ${P_i}^k$ is the k^{th} joint position in the i^{th} frame, ${P_r}^k$ is the referential marker group position relative to the k^{th} joint in the i^{th} frame, d is the constant distance between the marker group and actual joint, ε is the residual error.

Following up the solution of joint position, the next stage is to find the rigid body segments. Theoretically, the body segment between two adjacent joints should have the same length within all frames. Constant skeleton length for each rigid body segment can be computed over time. Ideally, the standard deviation

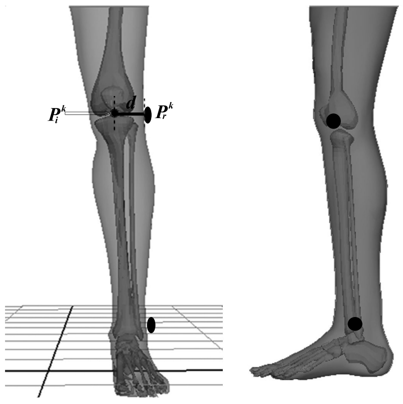


Fig. 2. Human character joint position calculation at knee joint

$\sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(l_i - \bar{l})^2}$ for each body segment is zero. Where l_i and $\bar{l}$ are length at i^{th} frame and mean of length over time respectively for each body segment.

3.2 Joint Angle Calculation

Once a hierarchical rigid body skeleton has been fitted according to the joint positions and rigid body segment, the joint's information in local and global coordinates are obtained based upon the recursive transformation matrix $X_w = R(q)X_l$. Where X_l, X_w are the joint position in local and global coordinates system respectively, and $R(q)$ is the recursive transformation matrix calculated from the global coordinates to current joint coordinates. The same recursive transformation matrix had been discussed in [15]. In our recursive method, the current i^{th} joint information within the fitted hierarchical skeleton is represented by its parent $(i-1)^{th}$ joint orientation matrixes. The i^{th} joint rotation angles are calculated by minimising the orientation error (Equation 2).

$$\Delta O(q) = \sum_{i=1}^{3}(P_i{}^* \cdot P_i(q) - 1)^2 \tag{2}$$

Where $P_i{}^*$(i = 1 to 3) are unit vectors along the desired joint orientation. $P_i(q)$ are the unit vectors along the current joint orientation. The representation of global and local coordinates of linked system is shown in Figure 3.

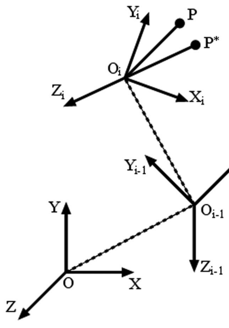


Fig. 3. Coordinates transformation of linked system. P is the current joint end position in i^{th} coordinates, P^* is the desired joint end position in i^{th} coordinates.

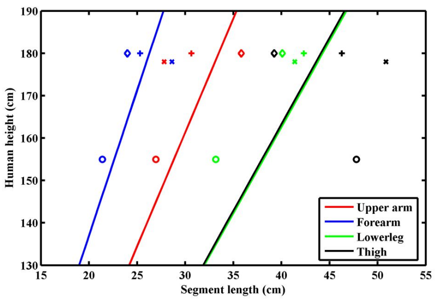


Fig. 4. Human body segment fraction of human height, + and × from two male performers, ∘ from a female. ◇ and + are from the same optical motion data, however, ◇ is obtained after the mapping procedure by the MotionBuilder software.

4 Implementation and Results

During the implementation, at the marker grouping stage, we automatically set markers group according to the prototype of human skeleton. For example, the four markers around the performer's waist will be used to define the root joint of the skeleton and the hip joints in the hierarchy. The constructed hierarchical skeleton includes 20 revolution joints. Each of them has three degrees of freedom, and the root joint has six degrees of freedom which are three rotations and three translations. In addition, the output of motion is the standard Biovision hierarchical data in which the rotation sequence of each joint is ZXY.

To infer the joint position, our method relies on the biomechanical information of human motion. The standard deviation for each body length over time is less than 2%. However, for a standard human skeleton, we are able to find an empirical formula for the calculation of the body segment length according to human heights [16]. In Figure 4, the calculated body segment lengths from two males and one female vs. standard anthropometry are shown. Moreover, one male dataset (marked as $\times$) are obtained according to the marker set of Vicon system. We also compared our data set with parameters obtained after the mapping procedure by an experienced motion capture producer using MotionBuilder software (marked as $\diamondsuit$). From the illustrated figure, we can see that there is obvious discrepancies between each data set. The biggest is the length of a female's upper leg. In fact, the female does have a much longer thigh compared with her height. Moreover, there are 1 to 4 centimetres differences measurement in other data sets compared with the anthropometry. These errors are due mainly to the fact that the marker shape in Reactor2 system is a 3 centimetres diameter disk. In order to prevent the marker moving from the attach point, the motion capture producer put the wrist crossbar and the ankle marker away from actual joint position about 1 to 2 centimetres to guarantee the marker position is stable. As long as the movement of hierarchical skeleton represents the movement of optical motion data, the mapping procedure is successful. For the final character animation, the skeleton motion will be edited for scene organisation, and the further motion retargeting is able to compensate the calculation error.

After the calculation and determination of the joint position and the length of each body part, our method is able to estimate joint orientations over time by our recursive method. In Figure 5, we show the original raw motion data and the constructed human skeleton. We noticed that the calculated position of the centre of mass of each individual rigid body segment is very much related with the extra marker used in Vicon system. Similar to the techniques presented in [15], at least three markers on one rigid body part are required to estimate the joint position. The method we presented for mapping optical motion data is not specific to Reactor2 system, it can fit other optical motion capture data as well. To test our method, we chose various optical motion data ranging from subtle motion to more dynamic movements, e.g., dancing and martial art.

All the tests were run on a 3.4Gz Pentium IV with 2GB of main memory. Since our skeleton joint orientation estimation is per frame based technology, the computation time for marker mapping and the skeleton joint calculation is

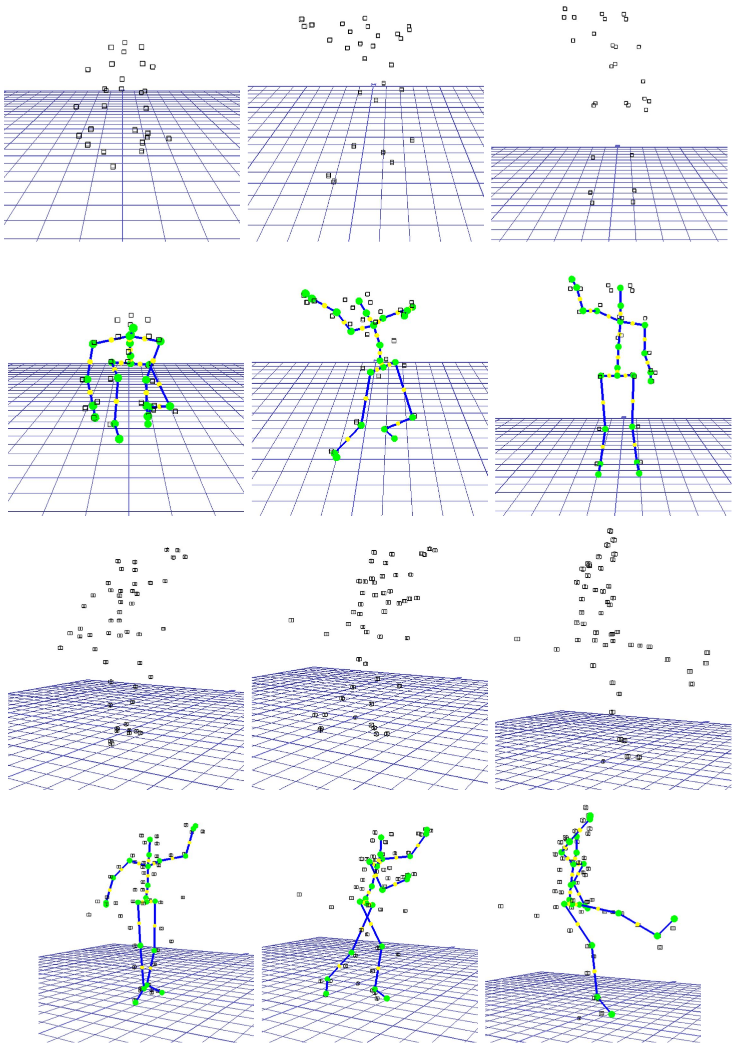


Fig. 5. Row 1 and row3 are the raw optical motion data based on each individual marker set captured from Reactor2 and Vicon system respectively (black squares show the marker positions). Row 2 and row 4 are the constructed skeleton (green spheres display the joints, blue bars are the rigid body segments, and the yellow dots show the centres of mass of each rigid body).

about 2ms, and for a simply stick character the graphic displaying is around 2ms. In respect of a complex geometric shape character, the time for graphics displaying may increase. Even though we have not tested complicated characters, we believe this off line automatic human-like skeleton joints estimation method can play on real time if the network streaming is fast enough.

5 Discussion

In this paper, we presented an automatic method for marker mapping and skeleton joint estimation from optical motion capture data. However, the accuracy of skeleton joint position and orientation relies heavily on the recorded marker positions. Since our method for joint angle calculation is per-frame based, there is a limitation for a motion capture system with a small number of markers. If one of the markers is lost or blocked, the signal detector would not record the marker position. The output of the skeleton motion could be unrealistic. In order to achieve satisfying motion data, the motion capture producer needs to either recapture motion data or clean up data after a capture session. The more information supplied from the optical markers, the more accuracy of the skeleton mapping can be achieved. After testing our method on other optical motion capture data, we noticed that the different naming conventions used by other optical systems would limit our automatic method. To get around this issue, we can redo the marker name mapping manually.

Acknowledgments

We would like to thank Andrew Cousins at AccessMocap motion capture studio for the Reactor2 motion data and for his valuable comments. Also, we would like to thank ATC for the Vicon system C3D format motion data.

References

1. Abe, Y., Liu, C.K., Popović, Z.: Momentum-based parameterization of dynamic character motion. In: Proceedings of Eurographics/ACM SIGGRAPH Symposium on Computer Animation, pp. 173–182 (2004)
2. Autodesk. Autodesk Motionbuilder. WWW Site (2007), `http://www.autodesk.com/`
3. Bodenheimer, B., Rose, C., Rosenthal, S., Pella, J.: The process of motion capture: Dealing with the data. In: Computer Animation and Simulation. Eurographics, vol. 97, pp. 3–18. Springer, Heidelberg (1997)
4. Glardon, P., Boulic, R., Thalmann, D.: PCA-based walking engine using motion capture data. In: CGI 2004: Proceedings of the Computer Graphics International, pp. 292–298 (2004)
5. Gleicher, M.: Retargetting motion to new characters. In: Computer Graphics, pp. 33–42 (1998); Proceedings of SIGGRAPH (1998)
6. Kirk, A., O'Brien, J.F., Forsyth, D.A.: Skeletal parameter estimation from optical motion capture data. In: Computer Vision and Pattern Recognition, pp. 782–788 (2005)
7. Kovar, L., Gleicher, M.: Flexible automatic motion blending with Registration Curves. Eurographics, 214–224 (2003)
8. LeVeau, B.F.: Williams and Lissner's Biomechanics of human motion, 3rd edn. W.B. Saunders Company, Philadelphia (1992)
9. Monzani, J.S., Baerlocher, P., Boulic, R., Thalmann, D.: Using an intermediate skeleton and inverse kinematics for motion retargeting. Eurographics 19(3) (2000)

10. Müller, M., Röder, T.: Efficient content-based retrieval of motion capture data. ACM Trans. Graph. 24(3); In: Proceedings ACM SIGGRAPH, pp. 677–685 (2005)
11. O'Brien, J.F., Bodenheimer, R.E., Brostow, G.J., Hodgins, J.K.: Automatic joint parameter estimation from magnetic motion capture data. In: Proceedings of Graphics Interface, pp. 53–60 (2000)
12. Popović, Z., Witkin, A.: Physically based motion transformation. In: Proceedings of SIGGRAPH, pp. 11–20 (1999)
13. Rose, C., Guenter, B., Bodenheimer, B., Cohen, M.: Efficient generation of motion transitions using spacetime constrains. In: Proceedings of SIGGRAPH, pp. 147–154 (1996)
14. Safonova, A., Hodgins, J.K., Pollard, N.S.: Synthesing physically realistic human motion in low-dimensional, behavior-specific spaces. ACM Trans. Graph. 23(3), 514–521 (2004)
15. Silaghi, M.C., Plänkers, R., Boulic, R., Fua, P., Thalmann, D.: Local and global skeleton fitting techniques for optical motion capture. In: Magnenat-Thalmann, N., Thalmann, D. (eds.) CAPTECH 1998. LNCS (LNAI), vol. 1537, pp. 26–40. Springer, Heidelberg (1998)
16. Winter, D.A.: Biomechanics and Motor Control of Human Movement, 2nd edn. Wiley, New York (1990)
17. Zhao, J., Badler, N.: Inverse kinematics positioning using nonlinear programming for highly articulated figure. ACM Trans. on Graph. 13(4), 313–336 (1994)
18. Zordan, V.B., Van Der Horst, N.C.: Mapping optical motion capture data to skeleton motion using a physical model. In: Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 245–250 (2003)

An Immersive Motion Interface with Edutainment Contents for Elderly People

HyungSeok Kim, YoungTae Roh, and Jee-In Kim*

Department of Internet & Multimedia Engineering, Konkuk University,
1 Hwayang dong, Kwangjin gu, Seoul, Korea
{hyuskim,mocamilk,jnkm}@konkuk.ac.kr

Abstract. Providing natural human-computer interaction has became one of the major issues as computer usage became popular. Starting from computer familiar users, consideration for non-familiar users including elderly and impaired users gets more attention recently. In this paper, a few approaches to provide natural interface for elderly people are summarized with a couple of experimental results. Our goal is to devise an immersive interface which is a platform providing the motion context. The interface is focused on edutainment contents, based on the observation that computer games for elderly people can be utilized as a tool not only for enjoying themselves but also for improving their health and quality of lives. In this paper, our approaches to use a tabletop interface, u-Table, and its corresponding natural gestures as a user interface for playing games are summarized. Since a table is a good place for people to get together, talk to each other and share their experiences, a tabletop interface can be served as a tool for playing games of cooperation, conversation and sharing. It was demonstrated that its users had better experiences with u-Table comparing to a mouse, a keyboard and a monitor.

Keywords: Tabletop Interface, Edutainment Contents, Immersive interface.

1 Introduction

Information technologies such as computers and the Internet become essential parts of our daily lives. We almost always use computers and access the Internet for working, studying, playing, enjoying, etc. To support those daily activities, it is necessary to provide *natural* interfaces to information system to users. Especially for handicapped people, current interfaces are not suitable in most cases.

New types of interfaces such as wands, data gloves, and gesture-based interface are experimentally used in virtual reality applications. Those interfaces are widely adopted in entertainment applications especially in arcade games and consol games such as Nintendo Wii.

However, even with those new technologies, it is not easy for elderly people to access and utilize computers and the Internet for their benefits of the modern information technologies [1].

* Corresponding author.

© Springer-Verlag Berlin Heidelberg 2008

In this position paper, we discuss roles of *natural* interactions in playing games especially for elderly people by summarizing our past and current efforts to devise new types of interfaces [10, 11].

Most of the elderly people did not have sufficient opportunities to be exposed to computer education. Also, user interfaces of current computers are not suitable for them to have such opportunities easily and conveniently. The combination of those factors could make them excluded from the new life styles based on information technologies [2].

The population of elderly people increases. It is expected that the percentage of the elderly people become 35% of the population of the whole world in 2050 [3]. It becomes an important social and economical issue how to help the elderly people enjoy information technologies with less problems and difficulties. Computer games could be an excellent starter for them. They could access computers and enjoy themselves with computer games during their spare times.

The problem is difficult and inconvenient user interfaces for them such as a monitor, a mouse and a keyboard. Therefore, the solutions must be provided with easy and convenient user interfaces. It is reported that the elderly people prefer to have multimedia interfaces [1].

With those observations, we consider two approaches.

- Providing familiar interaction metaphor for users
  By providing familiar interaction metaphor, it would decrease initial barrier to elderly people to use information system.
- Providing immersive interaction metaphor for users
  By providing immersive interaction metaphor, it would increase *believability* of experiences of elderly people.

With those arguments, we proposed to use a tabletop interface, "u-Table" [10], and its gesture-based interaction methods as a place for elderly people to play computer games. With u-Table, the elderly people could play and enjoy computer games which might not have been accessible and not have been enjoyable by them with conventional user interfaces such as a mouse and a keyboard. A collection of games were digitalized and we asked elderly people to play the games. Their responses and performances were quite satisfactory.

2 Related Works

A table is usually regarded as a central place at home and offices. While an individual can use a table, a group of people can also use it. People could meet, study, talk, eat, drink and play games together around a table. They can share information with their family, friends and colleagues over a table.

A tabletop interface, tele-communication technologies and virtual reality (VR) technologies were combined in order to provide elderly people with a virtual space for their family, relatives and friends living at remote locations [7]. The users can enter the virtual spaces through the tabletop interface, DiamondTouch developed by Mitsubishi. In order for accessing the virtual spaces, the DVE (Distributed Virtual Environment) system was used. Each participant of the system was represented by his avatar in the virtual space. They could play games, watch pictures and video tapes, sing songs together, etc. through the virtual space. It was expected that the segregated elderly people would enjoy the system and their mental health should be greatly enhanced.

Since most of elderly people are not familiar with traditional interaction devices such as a mouse and a keyboard, more *believable* interaction methods are required for them to play computer games.

The interaction process provides *believable* experiences if the interface is consistent and persistent within multi-modal virtual environments [13,14]. Among multimodal channels, visual and haptic has been considered primary channels in most cases of interaction. Due to this reason, there have been successful approaches to utilizing *tangible* interfaces for elderly people.

"Curball" is a collaborative computer game and uses a new interaction method for elderly people [8]. The game is a combination of Curling and Bowling. It uses balls and obstacles as its tangible devices for interacting with a computer. Both elderly people and young people can jointly participate in the inter-generation game by accessing its game server. The tangible devices and their associated gestures are used for throwing balls and arranging obstacles in the Curball game instead of a mouse, a joystick and a keyboard.

For elderly people, physical activities and exercises are quite important. However, the activities should not require intense and/or quick movements by considering their physical conditions. "Age Invaders" is a computer game for elderly people and children to play harmoniously in physical spaces [9]. They can communicate and play the game both physically and through the Internet. Physical players are on the game board and move physically to follow patterns on the board.

In comparison to "Age Invaders", most of the games are composed of interactions with visual information and haptic manipulations. To provide believable haptic interface it is necessary to provide persistent passive haptic feedback along with consistent active haptic feedback, if there exists. The passive haptic feedback is the feedback provided by the physical device itself, so that lies on persistently. The active haptic feedback is the feedback provided by active actions of the device, such as vibrations, to give 'intended' feedback. The 'u-Table' is proposed as one testbed in providing persistent haptic feedback [12].

In this paper, we provide recent experimental results in computer games for elderly people using a tabletop interface, u-Table. The games are played by multiple touches, gestures, and tangible interfaces for multiple players instead of a mouse, a joystick and a keyboard. Our experiments show that the elderly people prefer to play with u-Table rather than usual interfaces. The games include physical activities and mental activities which are supported by visual information given in familiar metaphor and persistent passive haptic feedback in believable way.

3 System Overview

We developed and tested some games for elderly people. The games are executed using a tabletop interface, u-Table, tangible objects and their associated interaction methods.

3.1 u-Table

The tabletop interface consists of a beam projector, a rear-projected screen, a frame, mirrors, a camera and a media server.[10]. Figure 1 shows an overall view.

The media server is the main engine for the games. The top of u-Table is a rear-projected screen which is used as not only an output display device but also an input device for recognizing fingertips and gestures of its users. Images from the beam projector are reflected by mirrors and displayed on the top of u-Table. Inputs from the users are recognized by the camera and the media server.

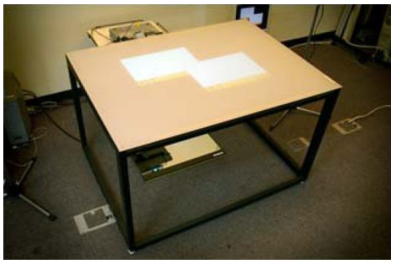


Fig. 1. u-Table

u-Table can be operated by fingertips and gestures which are intuitive and natural for elderly people to use. Therefore, elderly people hardly show reluctance to play computer games using u-Table. Also, this tabletop interface could be an excellent place to meet and play games together with friends, family and colleagues.

3.2 Interactions

The main interaction method of the games using u-Table is touching a screen with fingertips. The screen-touching method is used to select, drag and match widgets on the screen. The fingertips and the dragging gestures are the main interaction method.

The screen touching interaction provides natural mapping between visual information and action. In this way, the user can experience immersive motion, which is also defined as believable interaction.

Special fishing rods and fishing hooks were made for a game of fishing, The fishing rod gives feelings of throwing and fishing. The fishing hook is put on the screen and recognized by u-Table. When the users feel that fish can be caught using the fishing rod, they snatch the rod to get the fish.

4 Edutainment Contents

Four games were selected to be digitalized. Each one was presented in u-Table with appropriate tangible interface if necessary. When the digitalized game started, a brief introductory explanation was provided by both text and sound data. Like other games, cheerful sound and music were essential. Also, applauding messages and sounds were provided, when the users successfully finished the games.

Each game was designed to provide both physical exercises and mental exercised. Details of the desired effects are given in [11].

4.1 Virtual Fishing

Fishing is one of the favorite sports which could be played by elderly people. Its digitalized version using u-Table is called Virtual Fishing. The player could catch fish on a screen (a top of u-Table) using a special fishing rod. Sound effects and animation of fish were provided as feedbacks to users' actions of snatching a fishing rod, catching fish, finishing the game, etc.

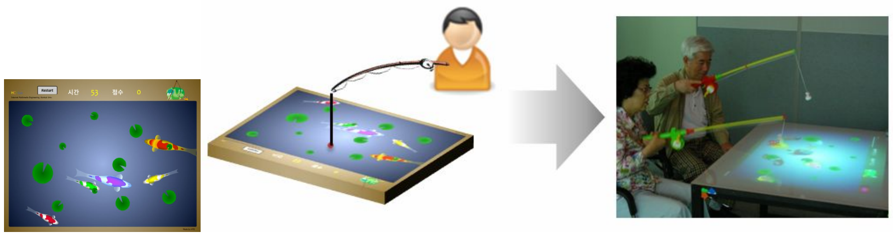


Fig. 2. Interaction Metaphor of Virtual Fishing

4.2 Word Puzzle

Elderly people liked to play with names and pictures of traditional items. The Word Puzzle game presented pictures of old items and their names on a screen. Colors and shapes of widgets on a screen could also be used in matching the items and the names as shown in Figure 3. Players could use their fingertips to select objects and drag them to combine.

Text, sound, and animation effects were provided, when meaningful events occurred. When a right pair of a picture and a word card was successfully combined, music and sound effects were provided.

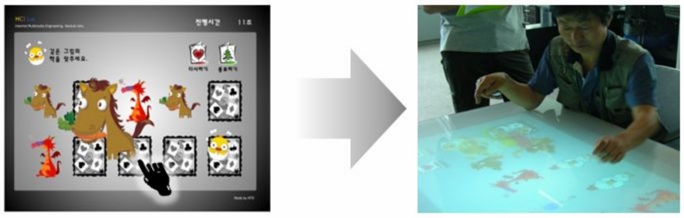


Fig. 3. Interaction Metaphor of Word Puzzle

4.3 Matching Cards

The "Matching Cards" game was known to be useful to maintain remembrance power of its players. First, the cards were open to be remembered by players for 30 seconds. Then, the cards were flipped. The players had to remember the cards and open to find a pair of matched cards.

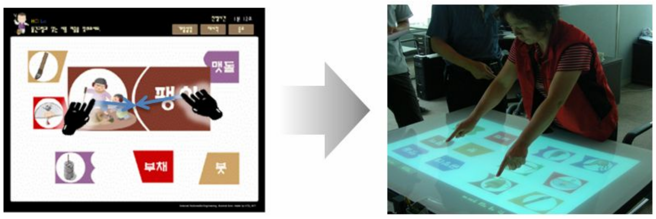


Fig. 4. Interaction Metaphor of Matching Cards

4.4 Bean-Pocket Game

The traditional bean-pocket game is throwing a bean/rice-packed 'pockets' to target balls. In the digitalized version, the traditional game is transformed into a kind of board game. User throws the 'pocket' onto the target which has a set of numbers on it. If the 'pocket' is placed onto a specific number, a stone corresponding to the user is moving forward on the board according to the number.

To increase interest, the game is designed based on a Korean fairy tale 'Sun and Moon'. The game background story is presented to the user before the game play as an introduction movie.

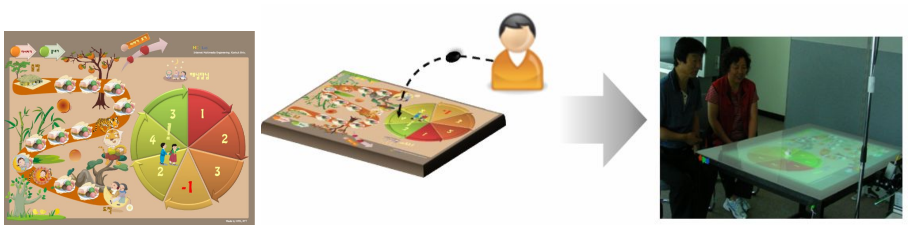


Fig. 5. Interaction Metaphor of Bean-Pocket Game

5 Experiments

5.1 Experiment 1: Familiar Interaction Metaphor

The experiment 1 is extended from the previous experiments [11]. The extension is made on evaluation the performance for the wider range of age groups. Also, Bean-Pocket game is added to the previous one.

The games were played by 15 people as the second study after the first one [11]. The subjects were equally selected for their age groups of 20, 30, 40, 50 and 60. Each age group consists of three women and three men.

To measure the effects of familiarity to the interaction metaphor, performances of playing the games were measured over trials. As the subjects executed the games, their performances such as numbers of caught fish and completion time were evaluated. To measure the effects of familiar interaction metaphor, we tested learning curves of the digitalized game.

5.1.1 Virtual Fishing

The subjects were asked to play Virtual Fishing for 1 minute. The numbers of fish caught by the players were counted after completing games. They played the game for three times. Figure 6 presents the numbers of fish caught by the different age groups for each trial. The numbers of fish increased in all age groups, as they tried more games.

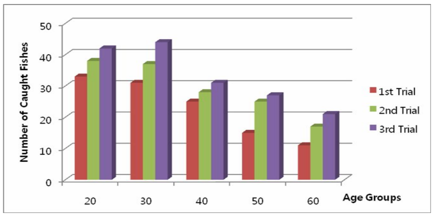


Fig. 6. Number of fishes caught in 1 minute by different age group for each trial

5.1.2 Word Puzzle

The completion time of Word Puzzle was presented in Figure 7. Each subject played it for three times. The familiarity of the interface could be detected, since the completion times of the subjects were reduced as they played more games.

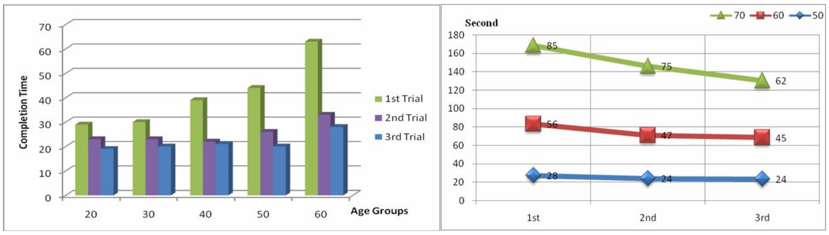


Fig. 7. Completion times of Games by different age groups Left: Word Puzzle game, Right: Matching Card game

5.1.3 Matching Cards [11]

Matching Cards game was tested in the pre-experiments with 6 subjects with three different age groups. In the extended experiment the game was removed because the types and result is quite similar with that of Word Puzzle. Both games require motion of both hands for the game with the card game metaphor. The pre-experiment result is shown in Figure 7. Each subject played it three times. In this experiment, the fifties group did not demonstrate the learning effects much. They seemed to be acquainted with u-Table quite quickly in this case of playing the Matching Cards game. It follows that u-Table is easier to learn and more convenient to use.

5.1.4 Bean-Pocket Game

Each player throws a pocket at the target on u-Table. Each subject played it three times after the pre-session. The pre-session consists of verbal explanation and a couple of trial throws.

The completion time for each trial is shown in Figure 8. At the first trial, most players were not able to throw the pocket correctly at the target even after the pre-session. But after the first game, most age groups showed the improved results.

As with the other games, this game shows good learning curves for most age groups. The players with age group 60 experienced difficulties in subtle motion control to throw the pocket precisely onto the target. But even for this case, the later trial gives much better performance than the first one.

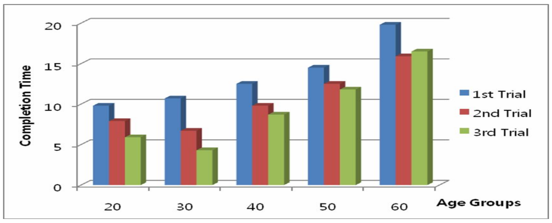


Fig. 8. Completion time of Bean-Pocket Games by different age groups

5.1.5 Subject Interview and Summary of the Experimental Results

The subjects were interviewed after the experiments. Their backgrounds and experiences with computers were examined. Feedbacks are summarized as following.

- Positive feedbacks
  - Was interesting to use u-Table and elderly groups preferred u-Table more.
  - Elderly groups showed more positive attitudes in computer games after the experiments
  - Exchanged opinion more with others
- Negative feedbacks
  - No active force feedback in u-Table
  - After hours of play, some muscle pain was reported.

Muscle pain is supposed to be caused by single-directional view for the entire session. Each subjects played 30 minutes or more for each game while focusing on the table with the same posture. For the better result it would be better to provide movement of whole body with view change, to reduce the undesired pains.

Mostly the interview feedback was positive. It is interesting to note that the u-Table is preferred more in the elderly groups, although the preference of u-Table is positive in all age groups. It could be assumed that younger groups are already familiar with keyboard and mouse and they could have better impression on it compared to the elderly groups who has little experience on keyboard and mouse interface.

In summary, it could be stated that the u-Table could provide good learning curves even for people with little experience of computers. Also, it is not very sufficient to state with only three trials, we can assume that if certain age group perceives the interface as familiar, it does not require excessive learning in interface but enables to be focused on the game itself. It can be deduced from the fact that for the higher age group, learning curve goes in steeper. The only exception is the Bean-Pocket game as it requires some physical limitation of motion control for elderly age group. In this case, learning curves of elderly groups show saturation behavior after a few trials.

The overall result could mean that even though the mental perception may require additional exercise while the physical perception could be learnt easily. This claim could be supported by the comparison experiments given in next section. The learning curves of conventional interface are steeper than that of u-Table [Figure 9].

As the further experiment, it is necessary to compare learning curves with more trials for different age groups. In this pilot experiment, results are clearly separated among different age groups. Thus we didn't apply any further statistical analysis on it.

5.2 Experiment 2: u-Table vs. Interaction without Immersive Motion [11]

It is assumed that the immersive motions provided by the believable metaphor with u-Table could provide better performances than the conventional interaction method with a mouse. To measure the effects of believable interaction metaphor, performance to complete each game was measured. If the performance and level of satisfaction in game is higher in proposed metaphor, it could be considered as more believable. This experiment is performed in our previous work [11]. In this section, summary and re-interpretation of results is given.

Total 8 subjects, 3 women and 5 men, were tested in this experiment. They were selected for three different age groups, 4 with their 50s, 2 of 60s and 2 of 70s. The subjects were asked to play the same games using a mouse and a wide-screen monitor as shown in Figure 10. Their performances were measured for the Word Puzzle game and the Matching Cards game. The completion times were measured. Bean-pocket game is not included in this experiment because the game itself is not suitable to be implemented using keyboard/mouse interfaces.

5.2.1 Performance Evaluation

As shown in Figure 10, the u-Table showed better performances. The subjects showed better performances with u-Table from the beginning to the end. It implies that u-Table is easier to learn and more convenient to use.

Most elderly people, who participated in the experiments, did not have previous experiences of using computers. From the beginning, they were reluctant to play the

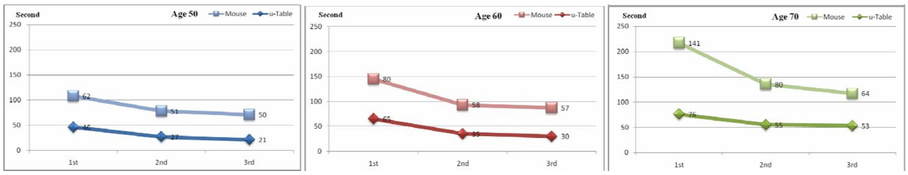


Fig. 9. Comparison of completion times of the Word Puzzle game for an interface with direct motion – u-table and an interface without a motion – mouse, by different age groups

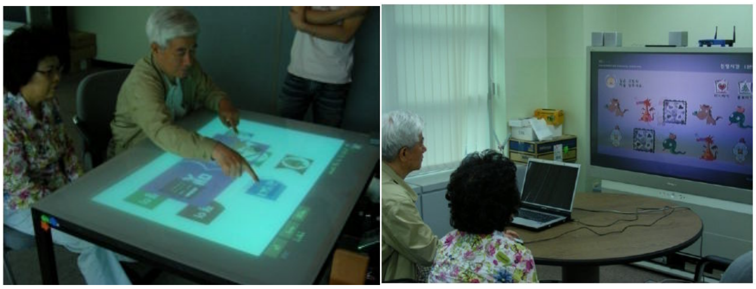


Fig. 10. Two types of play Left: Playing Word Puzzle with u-Table Right: Playing the Matching Cards game with a mouse and a wide-screen display device

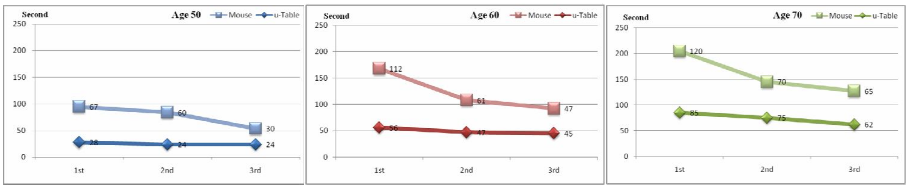


Fig. 11. Completion time of the Matching Cards game

games using a mouse. After they played the games using u-Table, they changed and tried the games with a mouse. Thus it could be safely considered that in learning curves for the mouse interface, the effect of learning the digitalized game itself is not included.

In the case of the Matching Cards game, the assumption was valid as shown in Figure 11. u-Table showed better performances than a mouse.

Two subjects had experiences with computers and the rest of the subjects never used computers before the experiments. The inexperienced subjects had a hard time to understand the relationships of a mouse and a cursor. They did not how to use a mouse and a screen smoothly in order to play the games at the beginning of the experiments. But they could learn and manipulate a mouse as they tried the games several times.

5.2.2 Subject Interview

The subjects were interviewed after the experiments. Their backgrounds and experiences with computers were examined. The subjective satisfactions about playing the games with u-Table were surveyed. The following questions are asked on the questionnaire.

1. Have you any experience in using computer?
2. Did you enjoy the game?
3. Did you feel any uncomfortness during play?
4. Was it fun enough in comparison with real-world games?
5. Which interface do you prefer as an easy game interface?
6. Which interface do you prefer as a comfort interface?
7. Do you change your mind on using computers after playing games with u-Table? If so, is it positive or negative?

Table 1. Answers to the questionnaire

Age	Gender	1	2	3	4	5	6	7
50	Man	○	○	○	○	u-Table	u-Table	Positive
51	Man	x	○	○	○	u-Table	u-Table	Positive
54	Woman	x	○	○	○	u-Table	u-Table	Positive
57	Man	x	○	○	○	u-Table	u-Table	Positive
63	Woman	x	○	○	○	u-Table	u-Table	Positive
65	Woman	x	○	○	○	u-Table	u-Table	Positive
70	Man	x	○	○	○	u-Table	u-Table	Positive
71	Man	x	○	○	○	u-Table	u-Table	Positive

All subjects were stated satisfaction with u-Table and the games. They told that they had changed their opinions about computer games. Also, the digitalized 'folk' games seemed to be friendlier to them rather than modern computer games. It means that with the proper contents and metaphor, the introduction to the computer could be easier than other methods. The interactions with fingertips were reported easier.

6 Concluding Remarks

In this paper, a tabletop interface, *u-Table*, and its associated tangible interfaces were applied to motion based games for elderly people. The interactions of the games were based on fingertips, tangible interfaces and their associated gestures. A set of games were digitalized and played by the elderly people.

They were asked to play the games with u-Table and a mouse for comparative experiments. The experimental results demonstrated that people could cooperate and talk over u-Table to complete the games. On the other hand, they played the games alone without much conversation, when the games were played using computers.

Through a series of experiments, we observed that the proposed interaction metaphor could be familiar and immersive, which brings better performances and more adaptive to the interface. The immersive motion interface could be a good alternative metaphor for edutainment contents for elderly people. To support this claim, we need to have more experiments, such as measuring complete learning curves with several more trials. Also, measurements of tiredness to each interface would be helpful to observe effect of believability.

It is planned to develop various kinds of games for elderly people. The main focus is to increase and/or at least maintain both mental and physical powers of elderly people. The games must be tested and evaluated with more elderly people.

A joint project will be proposed. A sanatorium and a university hospital will apply the u-Table based games to their patients. The goal is to develop a joyful and effective rehabilitation program for elderly people.

Acknowledgement

This research was supported by the Seoul R&BD Program (10581), 'Building an Innovation Cluster of Digital Contents Industries for u-Life'. The preliminary experiments were conducted in collaboration with TaeDoo Hwang, Jun Lee, SungJun Park and HaeWon Shin.

References

1. van de Watering, M.: The Impact of Computer Technology on the Elderly, Human Computer Interaction (2005), `http://www.few.vu.nl/~rvdwate/`
2. Heller, R., Jorge, J., Guedj, R.: Accessibility of Ubiquitous Computing: Providing for the Elderly. In: EC/NSF Workshop on Universal Accessibility of Ubiquitous Computing, May 22-25 (2001)

3. World Population Prospects: The 2004 Revision, Highlights, United Nations, New York, USA (2005)
4. Magerkurth, C., Engelke, T., Memisoglu, M.: Augmenting the Virtual Domain with Physical and Social Elements: Towards a Paradigm Shift in Computer Entertainment Technology. Computers in Entertainment (CIE) 2(4) (October 2004)
5. de Aguilera, M., Mendiz, A.: Video games and education. ACM Computers in Entertainment 1(1) (October 2003)
6. Gregor, P., Newell, A., Zajicek, M.: Designing for Dynamic Diversity, interfaces for older people. In: Proceedings of 5th ACM/SIGAPH Conference on Assistive Technologies, Edinburgh, Scotland, July 8-10, 2002, pp. 151–156 (2002)
7. Okano, Y., Ito, Y., Nitta, T.: A Study on the Application of DVE to a Mental Support System for the Aged Segregated from Family. In: TABLETOP 2006. IEEE, Los Alamitos (2006)
8. Kern, D., Stringer, M., Geraldine, Schmidt, A.: Curball – a prototype tangible game for inter-generational play. In: WETICE 2006. IEEE, Los Alamitos (2006)
9. Cheok, A.D., Lee, S., Kodagoda, S., Tat, K.E., Thang, L.N.: A Social and Physical Inter-Generational Computer Game for the Elderly and Children: Age Invaders. In: ISWC 2005. IEEE, Los Alamitos (2005)
10. Lee, J., Kim, J.-I.: u-Table: A Tabletop Interface for Multiple Users. In: Gavrilova, M.L., Gervasi, O., Kumar, V., Tan, C.J.K., Taniar, D., Laganá, A., Mun, Y., Choo, H. (eds.) ICCSA 2006. LNCS, vol. 3980, pp. 983–992. Springer, Heidelberg (2006)
11. Kim, J.-I., Hwang, T., Roh, Y., Lee, J., Park, S., Shin, H.: Application of a Tabletop Interface to Edutainment Contents for Elderly People. In: Tabletop 2007 (2007)
12. Kim, H.: Tangible multimodal interaction for ubiquitous media interface: touching believable media, presentation in La simulation virtuelle du toucher (October 2006)
13. Papagiannakis, G., Kim, H., Magnenat-Thalmann, N.: Believability and Presence in Mobile Mixed Reality Environments. In: Proceedings of IEEE VR 2005 workshop on virtuality structure, Bonn, Germany (March 2005)
14. Lee, J., Lee, J., Kim, H., Kim, J.-I.: Believable Interaction with a Quasi-tangible Tabletop Interface. Computer Animation and Virtual Worlds 18(2), 121–132 (2007)

Design of Experience and Flow in Movement-Based Interaction

Anton Nijholt, Betsy van Dijk, and Dennis Reidsma

University of Twente, Human Media Interaction,
PO Box 217, 7500 AE Enschede, the Netherlands
anijholt@cs.utwente.nl

Abstract. Movement-based and exertion interfaces assume that their users move. Users have to perform exercises, they have to dance, they have to golf or football, or they want to train particular bodily skills. Many examples of those interfaces exist, sometimes asking for subtle interaction between user and interface and sometimes asking for 'brute force' exertion interaction between user and interface. In these interfaces it is often the case that the interface mediates between players of a game. Obviously, one of the players may be a virtual human. We provide a 'state of the art survey' of such interfaces and in particular look at intelligent exertion interfaces, interfaces that know about their users and even try to anticipate what their users prepare to do. That is, we embed this interface research in ambient intelligence and entertainment computing research, and the interfaces we consider are not only mediating, but they also 'add' intelligence to the game. Other issues that will be discussed are 'flow' and 'engagement' for exertion interfaces. Intelligent exertion interfaces, being able to know and learn about their users, should also be able to provide means to keep their users engaged and in the flow of the game and entertainment experience. Unlike the situation with traditional desktop game research where we can observe lots of research activity trying to define, interpret and evaluate issues such as 'flow' and 'immersion', in movement-based interfaces these concepts need to be reconsidered and new ways of evaluation have to be defined.

Keywords: movement-based interfaces, exertion interfaces, ambient intelligence, games, entertainment, human-computer interaction, interaction coordination, design, evaluation.

1 Introduction

Nowadays, when we talk about human-computer interaction, it is not about the mouse and the keyboard anymore. Clearly, mouse and keyboard are useful and needed for many useful and boring tasks, but they do not provide natural and non-intrusive interaction between humans and the environments in which they live and work. The environments in which humans live are now becoming equipped with sensors that collect data about what is going on in the environments and are backed up by computers that integrate and interpret this data. Hence, we have environments that can observe their human inhabitants, can interpret what they know, want and do, and re-actively and

© Springer-Verlag Berlin Heidelberg 2008

pro-activily support them in their activities. In these ambient intelligence environments there is an inhabitant (often called a user), but more importantly, this ‘user’ is one of the many ‘agents’ that are modeled in such environments. Human inhabitants, (semi-) autonomous human-like agents (virtual humans, robots), and ‘intelligent devices’ such as furniture and other natural and obvious devices (pets, TV, pda’s ...) with embedded artificial intelligence will be considered part of these environments.

User interfaces have been introduced that offer, elicit and stimulate bodily activity for recreational and health purposes. Obviously, there are other applications that can be informed and guided by bodily activity information and that can be controlled by such information. For example, in a smart, sensor-equipped, home environment bodily activity can be employed to control devices, or the smart home environment might anticipate our activities and behave in a pro-active and anticipatory supporting way. Although in home environments there exists freedom concerning when and how to perform tasks, there are regular patterns of bodily activity and therefore activities can be predicted and anomalies can be detected. In task-oriented environments, e.g. an office environment, people probably have more well-defined tasks where efficiency plays an important role. Smart office furniture can provide context and task aware support to a moving office worker.

1.1 Exertion Interfaces

In previous years exertion interfaces have been introduced [1]. In game or entertainment environments the ‘user’ may take part in events that require bodily interaction with sensor-equipped environments. This can be a home environment, but it can be a city environment as well. For example, in a home environment we can have a user use an exercise bicycle or a treadmill to navigate or play a game in a ‘Second Life’-like environment. Clearly, we can inform the user about performance in the past (allowing him or her to compete with him- or herself) and we can inform the user about the performance of other users. In an urban game, mobile devices may be used to inform the users about activities they have to perform or about activities of their partners or opponents in the game. The game can require the gamer to walk, run, or perform other activities, in order to compete or cooperate with others involved in the game. Other types of exertion interfaces have been designed. Some characteristic examples will be discussed later in this paper.

In this paper we assume that exertion interfaces can be anywhere: in home, office, entertainment, sports, fitness, and medical environments, and also in public spaces. The motivation to use them can differ. For example, we can look at exertion exercises to improve health conditions, sports performance, or (therapeutic) physical rehabilitation. Often these interfaces are promoted from the point of view of fighting obesity. But, we want to look at exertion interfaces that are designed to provide fun and that engage a user in a game and entertainment experience, and in which considerations about health, physical performance, and rehabilitation are important, but by-products.

The main aim of this paper is to make an inventory of the issues that need to be considered when we want to embed exertion interface design in the frameworks that have been suggested for game design.

The organization of this paper is as follows. In section 2 we discuss existing exertion interfaces. A state-of-the-art survey is presented, where we look at exertion interfaces that allow direct and mediated interaction. Section 3 of this paper is on ‘intelligence’ in exertion interfaces. That is, how does the interface perceive and interpret the exertion activities of the user? Obviously, in general this requires interpretation of multi-modal input signals; in particular we look at audio-visual signals and the interpretation of these signals in order to provide the user with relevant (and stimulating) feedback. In section 4 we draw some conclusions and discuss future research.

2 Exertion and Entertainment Interfaces

As mentioned, in this paper we look at exertion interfaces. Exertion interfaces require exertion from the user. In fact, they are designed in order to elicit exertion. This requires an explanation. Reasons to design exertion interfaces are that exertion can be fun, social and satisfying (look at the many people that take part in sport events such as the New York, London, Berlin, or Rotterdam marathons) and that exertion interfaces can help users to improve their physical skills, and can help to improve health conditions.

In this section we give examples of various exertion interfaces. Some inventories of exertion interfaces were made by our students (see [2] and [3]) and in 2007 and 2008 two workshops were held at the Computer Human Interaction (CHI) conferences in San Jose and in Florence. Some of the examples mentioned below are drawn from these papers and workshops. We distinguish three ways of looking at exertion interfaces: adding game experience to exertion, adding exertion experience to games, and, obviously, have an approach where game, entertainment, and exertion experience come together in the design of an entertainment or game environment. Important is that in all these cases exertion and game elements are coupled. Game elements seduce and motivate users to engage in physical activity. Using exertion interfaces has also been called exergaming. Already in the early 1980’s we can recognize examples of exergaming, e.g. the Atari Puffer exercise bike or games with foot operated pads.

Clearly, an obvious way to obtain an exertion interface is to connect existing exercise devices (treadmills, rowing machines, exercise bikes) to an activity in a 3D virtual environment or in a game environment. The exercise device can be used to control a game, or to navigate in an interesting virtual environment (e.g., a beautiful landscape, or a Second Life city-like environment). In the virtual environment we can introduce challenges, competition and social interaction with other users. A well-known early example is the Virku (Virtual Fitness Centre) research project [4], where a traditional exercise bike is used to explore interesting surroundings and where environmental sounds are added to these surroundings to increase the presence of the user. Clearly, when a user cycles uphill it will take more effort and when going downhill less effort. In a similar project [5] it was investigated whether an increase in presence (by making the environment more realistic) led to an increase in performance. It turned out that the users not only pedaled faster, but also cycled much further without realizing how much more effort they put in.

In contrast to the idea of connecting an existing exercise device to a game or entertainment environment, we can also look at interfaces where ideas about exertion, games, and entertainment are there from the beginning of the design. One early example, the Nautilus game [6], can illustrate this. In this game a group of players have to work together and control the game (displayed on a big screen, sound effects and light effects) with the group's center of mass, speed and direction of movements that are detected by floor sensors. A more recent floor-sensor controlled game, where an existing game is provided with an exertion interface is a 'space invaders' game developed by the Mixed Reality Lab in Singapore [7]. In this game elderly and children play together and have to follow patterns that light up on the game floor, but they are also able to trigger bombs and rockets that force other players to jump out of the way and use other sub panels on the floor.

One of the best known exertion interfaces is 'sports over a distance', where players from different sites have to hit a wall with a ball [1]. The position on the wall and the force with which the ball hits the wall are mediated and made visible for opponents. A player can earn points by 'breaking' tiles on the wall and can profit from weak but not yet broken tiles that are left by his or her opponent. 'Sports over a distance' can be called a networked exertion interface. The same authors have introduced other networked exertion interfaces. For example, airhockey, table tennis, and, more recently, 'shadow boxing over a distance' [8].

In these latter applications entertainment, including social interaction has been the main reason to build these interfaces. Improving a particular skill in sports (e.g. baseball [9] or Tai Chi [10]) or improving fitness (aerobics [11] or physiotherapy [12]) have also been main reasons to introduce exertion interfaces. Finally, we need to mention the commercially available exertion interfaces. From the success of Dance Dance Revolution, Sony's EyeToy [13] applications and the WII Sports (tennis, golf, baseball, boxing and bowling), we now may expect to see more advanced exertion interfaces in the future, where the systems use more sensors that allow, among other things, audio-visual processing and interpretation of the user's activities and affective state.

An attempt to provide a taxonomy of exertion interfaces is presented in [14]. Recently a special issue on movement-based interaction appeared [15].

3 Intelligent Exertion Interfaces

Exertion interfaces require users to move their body and to use their arms and legs in order to get things done. Intelligent exertion interfaces understand what the user is doing and this understanding is used to provide better feedback.

Intelligent exertion interfaces detect a user's activity and the (possibly continuously) changing environment in which the user operates. That allows them to provide real-time feedback that displays understanding of what the user is doing and experiencing. This makes the difference between many of the current exertion interfaces and the advanced and intelligent interfaces that we see appear in research prototypes of exertion interfaces. In addition, dependent on the application, in intelligent exertion interfaces user feedback should be persuasive, motivating, and rewarding.

Game design requires designing game experience. We need to be aware which issues play a role in experience, how we can adapt them to a particular user during the game, and, in particular for our kind of research, what role does physical activity have in the game experience. One point in particular is what the user can tell us or the interface about his or her experiences. This can be done by conducting interviews, but rather we would like to see automatic detection of the user's experience and the automatic adaptation of the game or exertion environment to improve the experience.

3.1 More Advanced Sensing of User and Activities

As mentioned above, in order to design and implement successful exertion interfaces that know about the experience of the users, we need exertion environments that can detect, measure, and interpret physical activity. In the ball and shadow-boxing games of ‘sports over a distance’ [1,8], for example, there is no direct sensing of body movements or physiological information. ‘Only’ the result of the exertion (force, location) is measured and mediated. In contrast - without necessarily leading to a ‘better’ interface - there is also an interactive boxing interface, where the ‘punch’ is recognized using gesture recognition with computer vision [16,17].

Hence, there exist exertion interfaces with direct sensing of bodily activity (body movements, gestures, bodily and facial expressions, dynamic aspects of expression, etc.) and of speech activity that accompanies bodily activity (effort and pain utterances, laughs, prosodic aspects of speech utterances ...). Among the sensors are cameras and microphones that allow visual and audio processing of a user’s activity. They can provide information about location changes (tracking bodies and faces of individuals) and, among other things, frequency and expressiveness of movements. Other sensors in exertion interfaces can detect touch, pressure or proximity.

Sensing user’s activity in ambient entertainment environments is discussed in [18]. Rather than using questionnaires it is discussed how in the near future information obtained with computer vision and other sensors can help a movement-based interface to consider experience related issues such as personality, mood, and also pain, fatigue, frustration, irritation, etc.

One step further is to take into account physiological information obtained from the user. This information can be used both to guide the interaction and to measure the user experience [19,20]. In particular the recently started FP6 FUGA research project [21] is meant to find game experience measures that are based on psychophysiological recordings and brain imaging techniques. Clearly, BCI (Brain-Computer Interfacing) may be an extra source from which an interface can learn about the way the user experiences the interaction (besides using it to control the game as we can expect in the future) [22].

Finally, it should be mentioned that people will not always be willing to give away too much information about themselves, in particular when they see the computer or its embodiment in a virtual human as an opponent rather than as a system that tries to increase a positive gaming or exertion experience [18,23].

3.2 Exertion Interfaces: Flow and Immersion?

When modeling game experience the two issues that often arise are ‘flow’ and ‘immersion’. The theory of flow was introduced by Csikszentmihalyi [24]:

> “a sense that one’s skills are adequate to cope with the challenges at hand, in a goal-directed, rule-bound action system that provides clear rules as to how well one is performing. Concentration is so intense that there is no attention left over to think about anything irrelevant, or to worry about problems. Self-consciousness disappears, and the sense of timing becomes distorted. An activity that produces such experiences is so gratifying that people are willing to do it for its own sake, with little concern for what they will get out of it…”

Eight elements or features of this definition have been distinguished and generally it is assumed that these elements should be present in a game. They are: challenging activity that can be completed, facilitation of concentration, clear goals, immediate feedback, deep and effortless involvement, sense of control over one's actions, disappearing concern for the self, and finally, altered sense of duration of time. All these features can be found as prescriptions in present-day game design literature [25], sometimes using more refined features (agency, rewards, narrative …) and they play a role in game experience evaluation. Until now, they have hardly been explicitly considered in the design of movement-based interfaces for exertion and entertainment. A similar observation can be made for the concept of 'immersion'. Immersion [26, p.98] has been described as:

> "The experience of being transported to an elaborately simulated place is pleasurable in itself, regardless of the fantasy content. We refer to this experience as immersion. Immersion is a metaphorical term derived from the physical experience of being submerged in water. We seek the same feeling from a psychologically immersive experience that we do from a plunge in the ocean or swimming pool: the sensation of being surrounded by a completely other reality, as different as water is from air that takes over all of our attention our whole perceptual apparatus..."

Some attempts to provide more details to this definition can be found in the literature. For example, in [26] different types of immersion are identified: sensory immersion, challenge-based immersion and imaginative immersion. Levels of immersion (labeled engagement, engrossment, and total immersion) and how to cross barriers between these levels have been discussed in [27]. Finding out how to 'give hands and feet' to these concepts in movement-based interfaces and how to maximize flow and immersion in these interfaces are important research questions that need to be addressed in future exertion interface research. We can learn from some literature of flow in sports [28] and some recent and limited preliminary research on the design of exertion interfaces [29,30,31,32].

3.3 Multimodal, Joint, and Coordinated Activity in Exertion Interaction

Exertion interfaces emphasize the conscious use of bodily activity (jogging, dancing, playing music, sports, physical exercises, fitness, etc.) in coordination and sometimes in competition with other human users (friends, community or team members, accidental passers-by, opponents, etc.). Real-time coordinated interaction between human partners or between humans and virtual or robotic partners makes exertion interfaces exciting. In our research we are particularly interested in interfaces where the exertion interaction takes place with virtual or robotic characters or where the users are able to attribute human-like embodiment to the interface.

Coordination may be required by the rules of the game, the exercise or the tasks that have to be performed ask for it, but most of all people engage in coordinated interaction because it brings satisfaction and enjoyment. To illustrate this, the following citation is from Clark [33]:

> "A joint action is one that is carried out by an ensemble of people acting in coordination with each other. As some simple examples, think of two people waltzing, paddling a canoe, playing a piano duet, or making love."

Clearly, these are all joyful and engaging interactions. While Clark uses this observation to explore and develop theories of coordinated language use, we think it can be a useful observation when designing and evaluating exertion interfaces. We have studied face to face conversations, multi-party interaction, interactions between a virtual and a human dancer [34], a virtual conductor and a human orchestra [35], and a physiotherapist and her student [36] from the point of view of coordinated interaction [37].

Underlying joint activities are rules and scripts. To learn these and to put them into practice requires social intelligence, guided by empathy, moods and emotions. Despite many research results from social and behavioral sciences, computational models of joint activities are hardly available. This makes it difficult to design interfaces that aim at providing a similar interactional experience between real humans and virtual humans or robots, as is provided in a real-life human-human exertion activity, as in dancing, paddling, playing quatremains, and making love. Endowing the computer with a human-like appearance strengthens the expectation that the computer will take part in joint activities in human-like ways. Hence, there is a need for computational modeling of human joint activities. We replace one of the human partners in a joint exertion activity by a computer (i.e., a robot or a virtual human). Hence, we need to model joint exertion interaction in order to have the computer behave in a natural and engaging way.

In addition to rules that underlie joint activity there can be a need to align the interaction to external events over which the interaction partners do not necessarily have control. E.g., if we have a human and a virtual dancer then their moves have to be aligned with the music. Similarly, a virtual conductor and his human orchestra follow the score; a virtual aerobics trainer and human student have to align their movements to some kind of rhythm, often supported by upbeat music.

In our present research we investigate ways to measure engagement by looking at the degree of coordination between the activities of a human and a virtual partner in exertion and other entertainment interfaces [37]. In this research, supported by [38,39,40] we investigate how to make entertainment interactions more engaging by looking at interaction synchrony, where, on the one hand we aim at disturbing this synchrony in order to introduce new challenges, and on the other hand we aim at convergence towards coordinated anticipatory multi-modal interaction between human and artificial partners and their environment. Evidence that this approach will be successful is yet insufficiently available. Moreover, there are so many different types of exertion and movement-based entertainment interfaces that a comprehensive hypothesis about the role of interaction synchrony can not be expected to be given.

Design of experience and flow now receives much attention. Most research however is still about ways to characterize complex concepts such as experience, immersion, engagement, and flow. Exceptions are becoming available. For example, when we see the mentioning of ‘altered sense of duration of time’ in the description of flow, then indeed we can interview gamers about the time they think they have spent during a game with the actual time that has been measured. Interesting hypotheses related to our point of view on the role of interaction synchrony can be found in [41]. There the authors hypothesize that when players are immersed in a game their eye and body movements are different from those in a non-immersed situation. Obviously, again, there are many types of games and for video games where the gamer controls a game using mouse and keyboard or a joystick we have a quite different situation when the gamer is using a Wii remote control or a Wii Fit.

In our 'implicit' hypothesis on interactional synchrony we explicitly link this difference to the synchronization that is or is not present between gamer and game events. Notice that the main characteristic of 'flow' is the balance between challenges and skills. We can look at this as being able, as a gamer, to maintain a perfect coordination between eye, finger, and body movements on the one hand, and game/exercise events on the other hand. Obviously, when the game/exercise events are displayed by a virtual human, this coordination (or the disturbance of coordination to start up a new convergence of movements) becomes human-human non-verbal interaction coordination.

4 Conclusions

We surveyed characteristics of movement-based (or exertion) interfaces, i.e. interfaces that require and stimulate bodily activity. We discussed future research in this area by zooming in on sensor technology, intelligence and well-known game design and game experience principles. It was argued that for future development of interesting exertion (sports and entertainment) interfaces it is useful to embed this research in game design and game experience research. In addition we looked at a possible role for coordinated interaction research in the design and the evaluation of exertion interfaces.

Acknowledgments. This research has been supported by the Dutch National SmartMix project BrainGain on BCI, funded by the Ministry of Economic Affairs, and the GATE project, funded by the Netherlands Organization for Scientific Research (NWO) and the Netherlands ICT Research and Innovation Authority (ICT Regie).

References

1. Mueller, F., Agamanolis, S.: Exertion Interfaces: Sports over a Distance for Social Bonding and Fun. In: ACM Conference on Human Factors in Computing Systems (CHI 2003), pp. 561–568 (2003)
2. Bragt, J.: Do Exertion Interfaces Provide Better Exercise Environments? 2nd Twente Student Conference on IT, Enschede (2005)
3. de Koning, P.: Improving the Experience of Sports by adding a Virtual Environment. 3rd Twente Student Conference on IT, Enschede (2005)
4. Mokka, S., Väätänen, A., Heinilä, J., Välkkynen, P.: Fitness computer game with a bodily user interface. In: 2nd International Conference on Entertainment Computing, pp. 1–3 (2003)
5. IJsselsteijn, W., Kort, Y., de Westerink, J., Jager, M., de Bonants, R.: Fun and Sports: Enhancing the Home Fitness Experience. In: Rauterberg, M. (ed.) ICEC 2004. LNCS, vol. 3166, pp. 46–56. Springer, Heidelberg (2004)
6. Strömberg, H., Väätänen, A., Räty, V.: A group game played in interactive virtual space: design and evaluation. Designing Interactive Systems, 56–63 (2002)
7. Khoo, E.T., Cheok, A.D.: Age Invaders: Inter-generational mixed reality family game. The International Journal of Virtual Reality 5(2), 45–50 (2006)

8. Mueller, F., Agamanolis, S., Gibbs, M.R., Vetere, F.: Remote impact: shadowboxing over a distance. In: CHI 2008 Extended Abstracts on Human Factors in Computing Systems. ACM, New York (2008)
9. Komura, T., Kuroda, A., Shinagawa, Y.: NiceMeetVR: facing professional baseball pitchers in the virtual batting cage. In: ACM Symposium on Applied Computing, pp. 1060–1065 (2002)
10. Tan Chua, P., Crivella, R., Daly, B., Hu, N., Schaaf, R., Ventura, D., Camill, T., Hodgins, J., Pausch, R.: Training for physical tasks in virtual environments: Tai Chi. Virtual Reality, pp. 87–94. IEEE, Los Alamitos (2003)
11. Davis, J.W., Bobick, A.F.: Virtual PAT: A Virtual Personal Aerobics Trainer. MIT Media Laboratory, TR 436 (1998)
12. Babu, S., Zanbaka, C., Jackson, J., Chung, T.-O., Lok, B., Shin, M.C., Hodges, L.F.: Virtual Human Physiotherapist Framework for Personalized Training and Rehabilitation. In: Graphics Interface 2005, Victoria, British Columbia, Canada (2005)
13. Sony, Eye Toy: Kinetic (2005), `http://www.eyetoykinetic.com`
14. Mueller, F., Agamanolis, S., Powell, W., Nijholt, A.: Exertion Interfaces. In preparation
15. Larssen, A.T., Robertson, T., Loke, L., Edwards, J.: Special Issue on Movement-Based Interaction. Journal Personal and Ubiquitous Computing 11(8), 607–701 (2007)
16. Park, J.Y., Yi, J.H.: Gesture Recognition Based Interactive Boxing Game. International Journal of Information Technology 12(7), 36–44 (2006)
17. Höysniemi, J., Aula, A., Auvinen, P., Hännikäinen, J., Hämäläinen, P.: Shadow boxer: a physically interactive fitness game. In: Third Nordic Conference on Human-Computer interaction (NordiCHI 2004), vol. 82, pp. 389–392. ACM Press, New York (2004)
18. Nijholt, A.: Playing and Cheating in Ambient Entertainment. In: Ma, L., Rauterberg, M., Nakatsu, R. (eds.) ICEC 2007. LNCS, vol. 4740, pp. 415–420. Springer, Heidelberg (2007)
19. Picard, R.W., Vyzas, E., Healey, J.: Toward machine emotional intelligence: Analysis of affective physiological state. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(10), 1175–1191 (2001)
20. Picard, R.W., Daily, S.B.: Evaluating affective interactions: Alternatives to asking what users feel. Human Factors in Computing Systems. In: Workshop on Innovative Approaches to Evaluating Affective Interfaces (2005)
21. http://project.hkkk.fi/fuga/
22. Nijholt, A., Tan, D.: Playing with your Brain: Brain-Computer Interfaces and Games. In: Bernhaupt, R., Tscheligi, M. (eds.) Proceedings ACE (International Conference on Advances in Computer Entertainment Technology), pp. 305–306. ACM, New York (2007)
23. Nijholt, A.: Don't Give Yourself Away: Cooperation Revisited. In: Symposium Logic and the Simulation of Interaction and Reasoning at the AISB 2008 Convention Communication, Interaction and Social Intelligence, Brighton. The Society for the Study of Artificial Intelligence and Simulation of Behaviour, pp. 41–46 (2008)
24. Csikszentmihalyi, M.: Flow: the psychology of optimal experience. Harper & Row, New York (1990)
25. Sweetser, P., Wyeth, P.: Gameflow: a model for evaluating player enjoyment in games. Comput. Entertainment 3(3), 1–24 (2005)
26. Murray, J.H.: Hamlet on the Holodeck: The future of narrative in Cyberspace. MIT Press, Cambridge (1999)
27. Ermi, L., Mäyrä, F.: Fundamental components of the gameplay experience: analysing immersion. In: de Castell, S., de Jenson, J. (eds.) Changing views: worlds in play, Selected Papers DiGRA conference, Vancouver, pp. 15–27 (2005)

28. Jackson, S.A., Csikszentmihalyi, M.: Flow in sports: The keys to optimal experiences and performances. Human Kinetics, Leeds (1999)
29. Benford, S., Schnadelbach, H., Koleva, B., Gaver, B., Schmidt, A., Boucher, A., Steed, A., Anastasi, R., Greenhalgh, C., Rodden, T., Gellersen, H.: Sensible, sensable and desirable: a framework for designing physical interfaces. Technical Report Equator-03-003 (2003)
30. Consolvo, S., Everitt, K., Smith, I., Landay, J.A.: Design Requirements for Technologies that Encourage Physical Activity. In: ACM Conference on Human Factors in Computing Systems (CHI 2006), pp. 457–466 (2006)
31. Campbell, T., Fogarty, J.: Applying Game Design to Everyday Fitness Applications. In: ACM CHI 2007 workshop on Exertion Interfaces, San Jose, USA (2007)
32. Sinclair, J., Hingston, P., Masek, M.: Considerations for the design of exergames. In: 5th International Conference on Computer graphics and interactive techniques in Australia and Southeast Asia, pp. 289–295 (2007)
33. Clark, H.: Using Language. Cambridge University Press, Cambridge (1996)
34. Reidsma, D., Welbergen, H., van Poppe, R., Bos, P., Nijholt, A.: Towards Bi-directional Dancing Interaction. In: Harper, R., Rauterberg, M., Combetto, M. (eds.) ICEC 2006. LNCS, vol. 4161, pp. 1–12. Springer, Heidelberg (2006)
35. Maat, M., ter Ebbers, R., Reidsma, D., Nijholt, A.: Beyond the Beat: Modelling Intentions in a Virtual Conductor. In: 2nd International Conference on INtelligent TEchnologies for interactive enterTAINment (INTETAIN), ACM Digital Libraries (to appear, 2008)
36. Ruttkay, Z.M., van Welbergen, H.: On the timing of gestures of a Virtual Physiotherapist. In: Lanyi, C.S. (ed.) 3rd Central European MM & VR Conference, pp. 219–224. Pannonian Univ. Press, Hungary (2006)
37. Nijholt, A., Reidsma, D., Welbergen, H., van Akker, H.J.A., op den Ruttkay, Z.M.: Mutually Coordinated Anticipatory Multimodal Interaction. In: Esposito, A., et al. (eds.) Nonverbal Features of Human-Human and Human-Machine Interaction. LNCS, vol. 5042, pp. 73–93. Springer, Heidelberg (2008)
38. Michalowski, M.P., Sabanovic, S., Kozima, H.: A dancing robot for rhythmic social interaction. In: HRI 2007, pp. 89–96 (2007)
39. Tanaka, F., Suzuki, H.: Dance Interaction with QRIO: A Case Study for Non-boring Interaction by Using an Entrainment Ensemble Model. In: IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN 2004), pp. 419–424 (2004)
40. Tomida, T., Ishihara, A., Ueki, A., Tomari, Y., Fukushima, K., Inakage, M.: In: MiXer: the communication entertainment content by using "entrainment phenomenon" and "biofeedback". Advances in Computer Entertainment Technology, pp. 286–287 (2007)
41. Cairns, P., Cox, A., Berthouze, N., Dhoparee, S., Jennett, C.: Quantifying the experience of immersion in games. In: Cognitive Science of Games and Gameplay workshop at Cognitive Science (2006)

Relaxed Steering towards Oriented Region Goals

Ronan Boulic

VRLAB, Ecole Polytechnique Fédérale de Lausanne, EPFL
1015 Lausanne, Switzerland
Ronan.Boulic@epfl.ch
http://vrlab.epfl.ch

Abstract. This paper extends the *funnelling* behavior to offer a low-cost flexible guidance of mobile entities towards a circular region goal with the guarantee of enforcing an orientation within a predefined tolerance interval. The key requirements are the same as the funnelling control, i.e. a low and constant cost update of the control even when the goal parameters change (distance and relative orientation of the goal, position tolerance radius, orientation tolerance interval, desired speed). The smoothness and the optimality of the resulting trajectory being of high importance the paper qualitatively compares the trajectories produced by both funnelling algorithms. The new relaxed approach appears to produce smoother and shorter path for path made of a succession of large region goals. These qualities and its low cost advocate for its exploitation for moving through large dynamically changing regions without precise a priori planning.

1 Introduction

Despite describing a large repertoire of steering behaviors, the key paper from Reynolds [R99] does not offer a method for reaching a target position with a prescribed orientation. A fast controller achieving such a funnelling behavior has been introduced in [B05a,b] as illustrated in Fig. 1 that compares trajectories for reaching a target position (red circle) without (left) and with a prescribed orientation β (right). At its core lies a simple *seek* Proportional-Derivative orientation controller aiming at zeroing the orientation error α between the forward direction and the radial direction

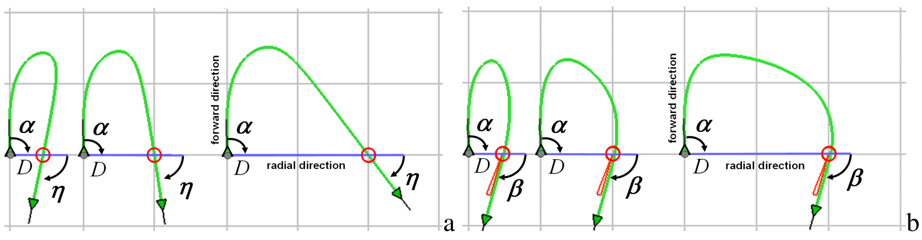


Fig. 1. : (a) reaching three position goals with a "seek" controller, (b) reaching the same position with a prescribed final orientation β with a *funnelling* controller [B05a,b]

© Springer-Verlag Berlin Heidelberg 2008

linking the current position to the desired position (Fig. 1 left). The funnelling behavior modulates the goal of the seek controller to achieve the desired orientation β.

The next section briefly situates this approach with respect to prior efforts in gait trajectory control for crowds with an emphasis on collision avoidance.

2 Background

Reynolds achieves collision avoidance in [R99] first by extrapolating the trajectory as a straight line; in a second stage a repulsive behavior is exploited whenever the extrapolated trajectory intersects an obstacle. This is generalized for multiple moving entities ; it results in a combination of turning away and accelerating/decelerating depending on the relative location of colliding entities [R00]. An especially demanding class of moving entities requesting finely tuned steering behaviors is the RobotCup soccer agent described in [BOM03]. This work stresses limitation of the Seek behavior such as getting trapped in local orbital minima and oscillations. Another important class of characters is pedestrians; Helbings and Molnar have reproduced the motion of pedestrians groups in some well-known contexts (e.g. corridor) by subjecting them to “social forces” resulting from the combination of potential fields [HM95]. Raup-Musse and Thalmann have exploited Bézier curves to introduce some variety in groups’ behavior within a crowd [R01]; collision avoidance is handled through the intersection of predicted linear trajectories. A similar approach of collision avoidance is adopted by Lamarche and Donikian [LD04]. Metoyer and Hodgins have associated potential fields to user-supplied natural path and proposed a higher level management of pedestrian collision avoidance [MH04]. Brogan and Johnson have built a walking path model from measurements from which they construct a heading chart ensuring trajectories with minimal radius of curvature towards a goal while avoiding static obstacles [BJ03]. Another approach based on measured data is described in [PPD07] where two subjects modify their trajectories to avoid colliding into each other. A model is derived for controlling walking agents in a crowd.

A planning approach exploiting probabilistic roadmaps is described in [P08]; the approach allows to prevent collisions between a moving human body and a complex static environment. A partial planning approach is proposed in Go et al. [GTK06]; it offers a good compromise between the reactivity to a dynamic environment and local minima avoidance. These authors pre-calculate 3D trajectory segments in a local coordinate system for a sampling of various initial conditions; this high precision prediction is made for the principal mobile entity while the prediction made for other mutually dependent vehicles exploit a linear extrapolation

The trajectories produced with the funneling behavior [B05a] show high similarities with the experimental trajectories studied in [ALHB06]. A comparison with Bezier curves highlights that their local curvature near the destination appears to be independent of the distance for distances over 4 meters [B05b]. The present paper examine how to relax the funnelling control so as to fully take advantage of the concept of *region goal*, for which any point of the region is equally valid as temporary goal. This kind of region goals can easily be inferred from higher level planning data structures such as the graph from [P08] or the corridors from [GKKO08]. We show that such an approach produces smoother and shorter paths than the standard funnelling approach. In the longer term we adhere to the idea of steering benchmarks that include not only relaxed funnelling reaches as described here but also more complex scenarios involving a higher management level for collision avoidance [SNKFR08].

The paper first illustrates the key concepts behind the standard funnelling approach and highlights the comparison with human trajectories. We then stress the limitation of the concept of position tolerance when exploited in that framework. The principle of the relaxed funnelling is then presented in details. Various case studies compare both algorithm prior to report on their relative computing cost. The conclusion summarizes the various advantages introduced by the relaxed funnelling and suggests future directions of researches.

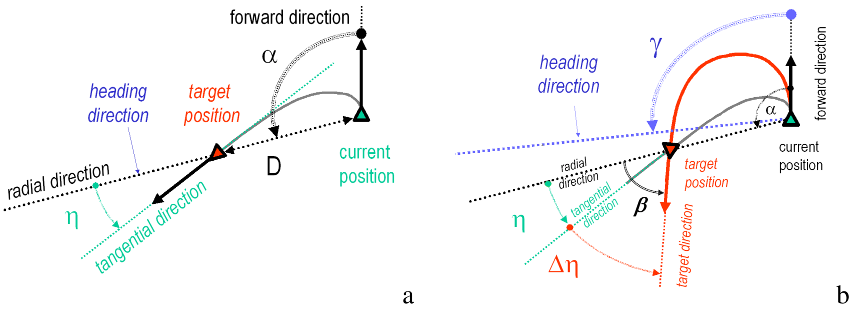


Fig. 2. (a) the radial direction is the heading direction for the seek controller, (b) funneling controller modulating the heading direction to achieve a target orientation

3 The Standard Funnelling Controller

The seek controller we exploit is different from the one proposed in [R99] as we prevent high angular and normal accelerations. The radial direction (Fig. 2a) is considered as the desired heading direction to be achieved by the mobile entity. One key angle to notice is the angle η made by the tangential direction at the target with the initial radial direction ; it is pre-computed for an anisotropic polar sampling of position goals (Fig 3a). The *funnelling* controller modulates the desired heading direction so that a desired final orientation β is finally achieved at the target position (Figure 2b). The modulation is a function of the angular difference $\Delta\eta=\beta-\eta$ between the desired direction β and the tangential direction η that would be obtained with the sole seek controller. This angle error $\Delta\eta$ is almost the only input for updating the angular acceleration ; it requests only to interpolate the η angle from the pre-computed table. In a second stage we ensure the convergence of the control algorithm by computing an upper limit for the quantity ($-\Delta\eta$) applied to the desired heading direction. The result is the *modulated heading direction* γ that is provided as instantaneous goal to the plain seek controller (Figure 2b).

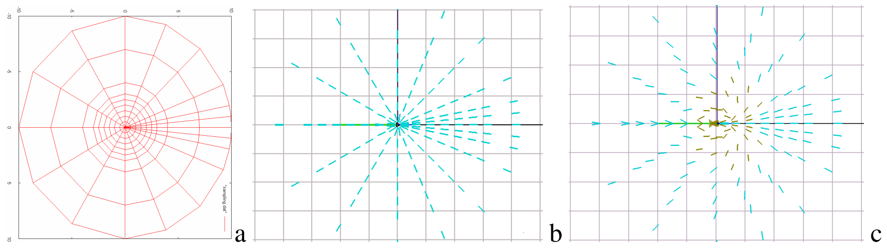


Fig. 3. (a) Anisotropic polar sampling for precomputing the η angle : the horizontal axis is oriented along the forward direction of the mobile entity. Central region of the local funnelling field build from the precomputed η angles for slow (b) and normal (c) linear velocities (null angular velocity in both cases). The darker colour around the mobile entity in (c) indicates that the corresponding sampled target could not be reached with the default seek controller [B05a]; the η angle obtained successfully for a smaller desired velocity is stored and displayed instead.

Conceptually the funnelling approach can be viewed as a field-driven steering approach. A major difference with other approaches relying on vector fields for steering is that our field is expressed in the local coordinate system of the mobile entity. Moreover the stored η angle for each sampled goal (Fig 3a) is a function of three variables, namely the current linear and angular velocities and the desired linear velocity. This explains the displayed variations between Fig.3 b (slow linear velocities) and Fig. 3c

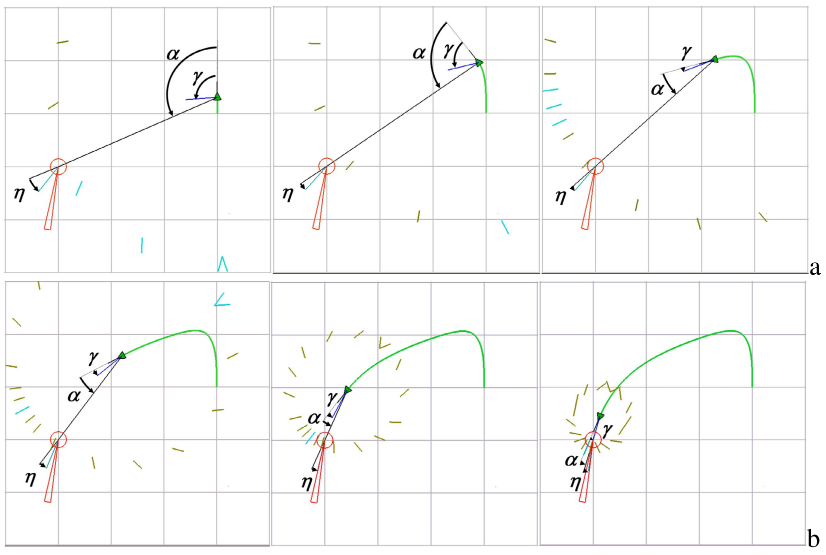


Fig. 4. (a -> b, left to right) examples of states extracted from the trajectory of a case similar to Fig2b where the target location and orientation are indicated by their respective tolerances with a circle and a triangle. Also visible are the current radial direction in black, the interpolated η angle at the target location, the modulated heading direction γ at the current location and the closest_distance ring of the local field.

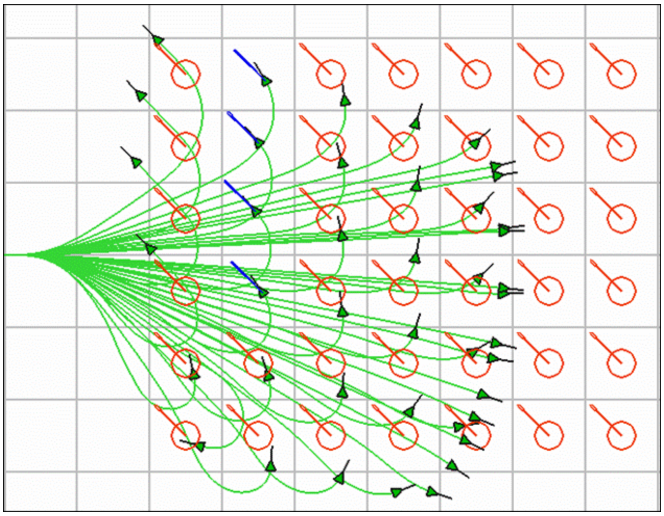


Fig. 5. Funnelling trajectories replicating human gait trajectories measured in [ALHB06]

(normal desired linear velocities). In the first case the final tangential directions appear to be oriented radially (Fig. 3b) whereas this is only the case for sampled goals lying in front of the mobile entity for Fig.3 c.

Fig. 4 a,b show both η and γ angles displayed for states extracted from a case analog to the one of Fig2. b. Only one ring of the local vector field is displayed for clarity purpose. In that example the initial and desired linear velocities have mid-range values. More trajectories resulting from the funnelling control are shown on Fig.5. All have the same initial state and desired global orientation but distinct target locations. This case corresponds to one case of the experimental protocol used in [ALHB06] for measuring human gait trajectories. The simulated funnelling trajectories offer high similarities with those captured [ALHB06]. For more details on the standard funnelling see [B05a,b].

4 Funnelling towards a Region Goal

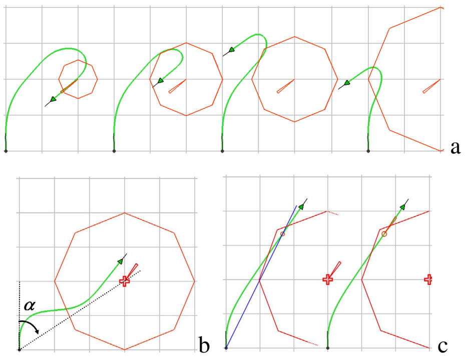


Fig. 6. Relaxed steering, (a) (b) with a tolerance, (c) sampled goal with smallest $\Delta\eta$

Defining a tolerance around the desired position (red circle) allows to relax the steering to a pure direction control once the position tolerance is met (Fig. 6a). However in these examples the α angle is still computed with respect to the centre of the region (Fig. 6b).

In the present section we introduce a relaxation of the target location in order to produce a smoother trajectory (e.g. Fig. 6c).

4.1 Tolerance Management vs. Region Goal

The relaxation introduced by the concept of *region goal* goes beyond the basic exploitation of the position tolerance in the standard funnelling algorithm. Let us recall that standard approach first; in that context, whenever the current location of the mobile entity appears to lies within the circle defined by the position tolerance, the control does not care anymore about the position control but focus only on completing the desired orientation [B05a]. This explains why in Fig 6a the mobile entity does not always achieves *simultaneously* the desired location and the desired orientation. We advocate for adopting such flexible approach as we can always reduce the radius of the position tolerance to enforce a stricter achievement.

Despite the flexibility offered by such a tolerance management, the control is still based on the location of a single point within the tolerance circle, i.e. its centre. This may lead to artificially curved trajectories as illustrated on Fig. 6b. One intuitively understands that, within the tolerance circle, there may exist a better target location for achieving the desired orientation. This is illustrated on Fig. 6c where indeed a different target position induces a smoother trajectory while respecting both the desired position and the orientation tolerances.

Thus the *relaxed funnelling* is based on the concept of *region goal* for which any point belonging to the tolerance circle is eligible to become the target position used for the algorithm described in the previous section. Then two questions arise: how to select such a point within the tolerance circle ? and how do we ensure the stability of such a relaxed steering ?

4.2 Selecting the Best Target Position and Ensuring Stability of the Control

4.2.1 Principal Criterion

Selecting the best target position means defining an optimality criterion. We have logically adopted the criterion of minimizing the $|\Delta\eta|$ quantity within the tolerance disk as justified now. This choice derives from the way the standard funnelling control is defined for achieving a desired orientation. Indeed when the desired orientation β matches the η angle then the $\Delta\eta$ quantity is null and the funnelling control reduces to a simple seek strategy. Our working hypothesis is that such a seek control defines a control cost minima compared to all other cases for which $|\Delta\eta| > 0$. So selecting the best target position amounts to find the point(s) within the tolerance circle that minimize $|\Delta\eta|$ because such a target location minimizes the control cost.

4.2.2 Secondary Criterion

The second issue to address is whether multiple optimal points exist in the sense of the above criterion and how to choose the best one among them. Fig. 3 provides hints that indeed there is often a complex subspace of the tolerance region that minimizes the $|\Delta\eta|$ criterion. As a consequence we have considered an additional secondary criterion of minimizing $|\alpha|$ that is exploited only for the solutions producing a null $|\Delta\eta|$. Minimizing such a secondary criterion corresponds to favor straight line movements over curved movements which is particularly pertinent for producing smooth and short trajectories.

4.2.3 Restricted Search of the Optimal Relaxed Direction

Even the two-levels criterion outlined above may produce multiple optimal solution, e.g. when the tolerance region lies in front of the mobile entity. For this reason, and with the additional requirement of minimizing the search computing cost, we have retained to restrict the search to the arc $\mathcal{A}= [\alpha_{min}, \alpha_{max}]$ defined as the intersection of the position tolerance disk with the sampled arc of radius nearest to the disk centre. Fig. 4 displays the full nearest ring for a few states along a given trajectory towards an oriented goal (only the vectors built from the sampled $\alpha + \eta$ quantity are displayed, hence their number is reduced for a distant target as visible on Fig. 3a).

4.2.4 Determining the Optimal Relaxed Direction α_r

The second major working hypothesis for determining the optimal relaxed direction is the monotonous variation of $(\alpha + \eta)$ over the whole restricted arc $\mathcal{A}$ even if it crosses 0; by construction the relaxed steering handles only cases with $\mathcal{A}$ included within [-π,+π].

This hypothesis ensures that we have a monotonous span of final orientation $\mathcal{F} = [\alpha_{min}+\eta(\alpha_{min}), \alpha_{max}+\eta(\alpha_{max})]$ over the restricted arc $\mathcal{A}$. The interval $\mathcal{F}$ itself may spread beyond [-π,+π]. The relaxed steering algorithm then boils down to find the angle α_r within $\mathcal{A}$ that minimizes the two-levels criterion described above between $\mathcal{F}$ and the desired orientation tolerance interval $\mathcal{T}$. The pseudo-code is given in Fig. 7.

```
if the centre of $\mathcal{F}$ and $\mathcal{T}$ have opposite signs
  if $\mathcal{F}$ and $\mathcal{T}$ overlap partially
    $\Delta\eta$=0, $\alpha_r$ = 0, **End**
  else
    Determine {{overlap},{gap}} for { $\mathcal{F}, \mathcal{T}, (\mathcal{T} \pm 2\pi)$ }
  end_if
else //the centre of $\mathcal{F}$ and $\mathcal{T}$ have the same sign
  if one boundary of $\mathcal{F}$ and of $\mathcal{T}$ have the opposite sign
    $\Delta\eta$=0, $\alpha_r$ = 0, **End**
  else
    Determine {{overlap},{gap}} for { $\mathcal{F}, \mathcal{T}$ }
  end_if
end_if
if some overlap exist
  $\Delta\eta$= 0,
  $\alpha_r$ = mapped $\mathcal{A}$ value of the smallest |overlap boundary|
else
  $\Delta\eta$= smallest gap between $\mathcal{F}$ and other intervals
  $\alpha_r$ = $\mathcal{A}$ boundary of the smallest |gap|
end_if
**End**
```

Fig. 7. Pseudo-code for determining the optimal relaxed direction α_r

The algorithm complexity is independent from the size of the intervals as we check only the boundaries (owing to the monotony hypothesis). This makes the relaxed steering computing cost similar to the one of the standard funnelling. We choose to reduce the cost of evaluating the two η values for $\mathcal{A}$ boundaries by performing a simple linear interpolation from the table instead of a quintic linear one.

4.3 Two Examples

Figure 8 illustrates how the case of Fig6b is treated with the relaxed funnelling algorithm ; the modulated heading direction γ is constructed with respect to the relaxed direction α_r (to be compared with the standard radial direction α also shown).

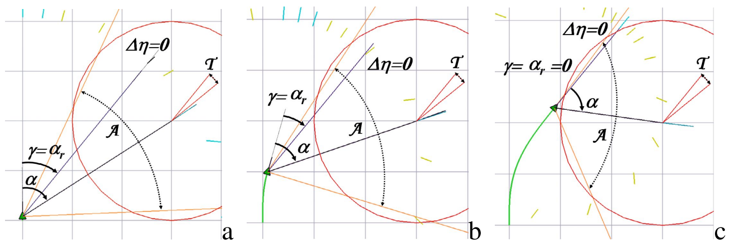


Fig. 8. The relaxed funnelling minimizes $|\Delta\eta|$, and $|\alpha_r|$ in case $\Delta\eta$ is null

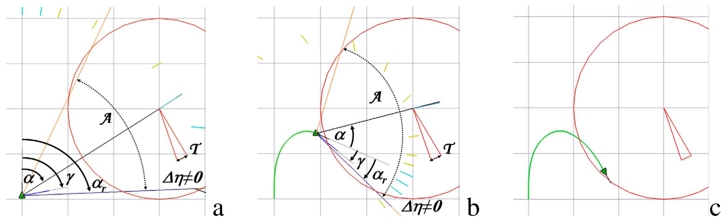


Fig. 9. $\Delta\eta$ is not null so only the $|\Delta\eta|$ minimization criterion is exploited to determine α_r

Figure 9 illustrates a more difficult case for which no relaxed direction can produce a null $\Delta\eta$. In that case the boundary of $\mathcal{A}$ minimizing $|\Delta\eta|$ is retained for α_r.

5 Trajectory Comparison

We present the trajectories obtained for the standard funnelling and the relaxed version introduced in this paper for the same set of moving entities, oriented targets and tolerances. Fig. 10,11, and 12 all show the resulting movement for nine mobile entities starting on the left side of the figures with an initial horizontal orientation. Fig. 10a,b,c are obtained with the standard funnelling whereas Fig.10 d,e,f are obtained with the optimal relaxed direction. The target is defined by the tolerance disk with a desired orientation indicated by the needle in the centre. Once the target is reached the mobile entities continues their path towards the next target (it was set as much as possible in the same direction as the desired orientation). The comparison of a with d, b with e, and c with f clearly show that the relaxed steering produces less curved trajectories for the same mobile entities. The traveled distance also appears to be frequently smaller.

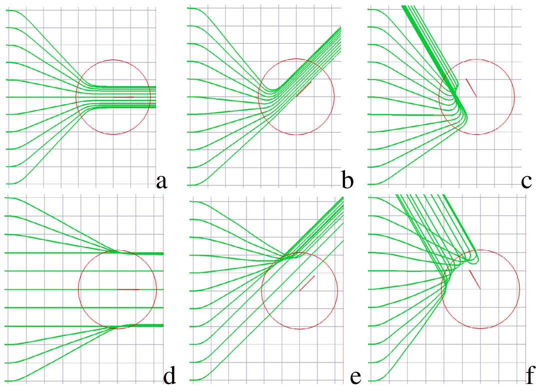


Fig. 10. Trajectories produced by the standard funnelling (top row) and the relaxed funnelling (bottom row) for the same initial conditions: nine mobile entities enters the image from the left side ; they all have the same position goal visible with a large tolerance circle of about 2m radius with a prescribed orientation indicated by a pin shape(a & d: 0°, b & e:45°, c & f:120°). After reaching this goal they all have the same distant goal (not visible).

Fig. 11 highlights that the orientation tolerance is mostly beneficial to the relaxed steering approach. Fig. 12 links multiple successive oriented targets. The relaxed steering appears to produce a globally smoother and shorter path.

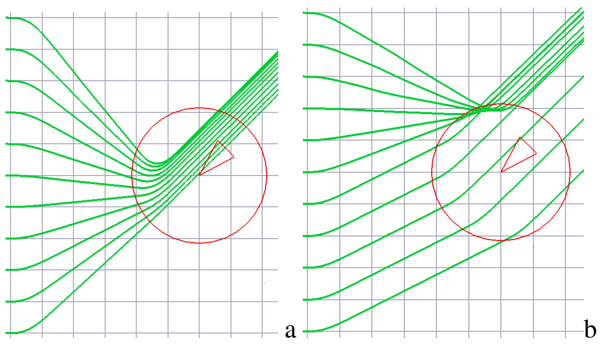


Fig. 11. Increasing the orientation tolerance has a minor impact on the trajectories produced with the standard funnelling (a) compared to the much wider spread of trajectories achieved with the relaxed one (b).

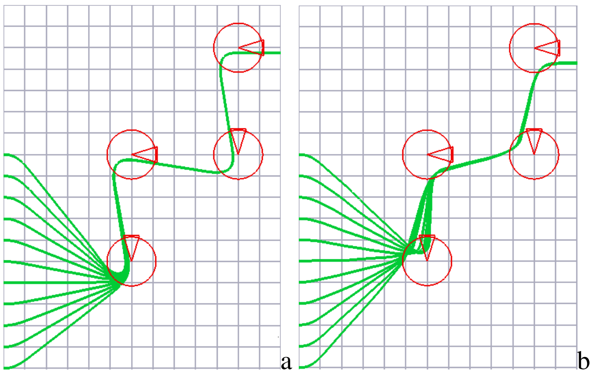


Fig. 12. Successive goals highlights the longer and more curved path produced by the standard funnelling (a) compared to the more globally smoother and shorter one obtained with the relaxed algorithm (b). Note too the tendency of trajectories to converge to a single path independently of the initial conditions.

7 Performances and Conclusion

The funnelling control update cost (i.e. per frame) is almost independent of the goal parameters. Measurements have been performed on an IBM Thinkpad T60 with Intel Centrino Duo, 1.83 GHz. The computing costs (without display) are reported in Table 1 for the release executable code produced with Microsoft VS 2005 (the algorithm implementation can still be improved). Average values have been computed over roughly 100000 iterations of each algorithm.

Table 1. [min, max] measured average computing costs of one funnelling control update in µs (with the corresponding frequency in KHz) for a large range of oriented goal parameters

	Small tolerances (0.1m and 1°).	Large tolerances (2.5m and 15°)
Standard funnelling [B05]	[11µs, 38µs] (90 KHz, 26 KHz)	[20µs, 39µs] (50 KHz, 25 KHz)
Relaxed funnelling	[15µs, 26µs] (67 KHz, 38 KHz)	[8µs, 26µs] (125 KHz, 38 KHz)

Table 1 highlights that the standard and the relaxed funnelling costs do not differ significantly when small position and orientation tolerances are used (an example of application of such a context is handling precise interactions with objects of the environment). On the other hand the relaxed funnelling tends to be on average 50% cheaper for large tolerances typical of outdoor wandering with no precise path to follow. As a side note, the animation files that are visible on the associated web site (http://vrlab.epfl.ch/~boulic/Steering/index.html) have been slowed down for easing the understanding of the algorithm.

By construction the relaxed funnelling can smoothly integrate standard collision avoidance approaches by adding the heading direction modulation induced by the collision avoidance to the one resulting from the funnelling (either standard or relaxed). In case the contribution due to the collision avoidance pushes the mobile entity away from its goal, the relaxed funnelling is able to dynamically adjust the optimal relaxed direction within $\mathcal{A}$ at the next frame.

To conclude, we have presented the relaxed steering that monitors whether an easier to reach goal exists within the tolerance region and maintain always the easiest to reach goal direction. Its relatively low cost make it a good candidate for handling a large number of moving entities in a dynamically changing environment. Future work will first evaluate the performance of the relaxed steering when combined with collision detection and avoidance. We also plan to introduce the possibility to handle lateral displacements within the general steering approach.

Acknowledgements

Thanks to Achille Peternier for the MVisio viewer.

References

[ALHB06] Arechavaleta, G., Laumond, J.-P., Hicheur, H., Berthoz, A.: Optimizing principles underlying the shape of trajectories in goal oriented locomotion for humans. In: 6th IEEE-RAS International Conference on Humanoid Robots, pp. 131–136, December 4-6 (2006)

[BOM03] Ben Amor, H., Obst O., Murray, J.: Fast, Neat and Under Control: Inverse Steering Bahaviors for Physical Autonomous Agents, Research Report, Institüt fèr Informatik, Universität Koblenz-Landau (December 2003)

[B05a] Boulic, R.: Proactive Steering Toward Oriented Targets. In: Proceedings of Eurographics 2005 Short Presentations, Dublin (2005)

[B05b] Boulic, R.: Reaching Oriented Targets with Funnelling Trajectories. In: Proc. of V-CROWDS 2005, Lausanne, November 24-25 (2005) ISBN 10 2-8399-0118-8

[BJ03] Brogan, D.C., Johnson, N.L.: Realistic Human Walking Paths. In: Proc. of Computer Animation and Social Agents, pp. 94–101 (2003)

[GKKO08] Geraerts, R., Kamphuis, A., Karamouzas, I., Overmars, M.: Using the Corridor Map Method for Path Planning for a Large Number of Characters. In: Egges, A., Kamphuis, A., Overmars, M. (eds.) MIG 2008. LNCS, vol. 5277, pp. 11–22. Springer, Heidelberg (2008)

[GTK04] Go, J., Thuc, V., Kuffner, J.J.: Autonomous Behaviors For Interactive Vehicle Animations. In: Proc. of Eurographics/ACM SIGGRAPH Symposium on Computer Animation, pp. 9–18 (2004)

[HM95] Helbings, D., Molnar, P.C.: Social force model for pedestrian dynamics. Physical Review E 51(5), 4282–4286 (1995)

[LD04] Lamarche, F., Donikian, S.: Crowds of Virtual Humans: a New Approach for Real Time Navigation in Complex and Structured Environments. Computer Graphics Forum, 23(3) (2004); In: Eurographics 2004

[MH04] Metoyer, R.A., Hodgins, J.K.: Reactive Pedestrian Navigation from Examples. The Visual Computer 20(10), 635–649 (2004)

[P08] Pettré, J.: Populate your Game Scene. In: Egges, A., Kamphuis, A., Overmars, M. (eds.) MIG 2008. LNCS, vol. 5277, pp. 33–42. Springer, Heidelberg (2008)

[PPD07] Paris, S., Pettré, J., Donikian, S.: Pedestrian Reactive Navigation for Crowd Simulation: a Predictive Approach,Computer Graphics Forum. In: Eurographics 2007, vol. 26(3), pp. 665–674 (2007)

[PVT08] Peternier, A., Vexo, F., Thalmann, D.: The Mental Vision framework: a platform for teaching, practicing and researching with Computer Graphics and Virtual Reality. Transactions on Edutainment 1 (June 2008)

[RT01] Raupp Musse, S., Thalmann, D.: Hierarchical Model for Real Time Simulation of Virtual Human Crowds. IEEE Transactions on Visualization and Computer Graphics 7(2), 152–164 (2001)

[R99] Reynolds, C.W.: Steering Behaviors For Autonomous Characters. In: Proc. 1999 Game Developer's Conference, pp. 763–782 (1999)

[R00] Reynolds, C.W.: Interaction with Groups of Autonomous Characters. In: Proc. 1999 Game Developer's Conference (2000)

[SNKFR08] Singh, S., Naik, M., Kapadia, M., Faloutsos, P., Reinman, G.: Watch Out! A Framework for Evaluating Steering Behaviors. In: Egges, A., Kamphuis, A., Overmars, M. (eds.) MIG 2008. LNCS, vol. 5277, pp. 200–209. Springer, Heidelberg (2008)

Opening Doors in Motion Analysis Research

Arjan Egges

Center for Advanced Gaming and Simulation
Utrecht University, Utrecht, the Netherlands
egges@cs.uu.nl

Abstract. This paper discusses the use of motion analysis techniques to establish a computational model of human motion. As a test case, we discuss an experiment that analyzes combined navigation and manipulation motions performed by human subjects. In this particular experiment, we have analyzed the action of walking to a door and opening the door in different settings. We have looked at different variables in this experiment, such as the body part that serves as the steering wheel, gender, and handedness. The results from this experiment form the first step toward a computational model of human motion.

1 Introduction

Virtual environments are extremely popular in a many fields, such as the game industry, social networks and virtual training programmes. The perceived realism and quality of these environments not only depends on the quality of the 3D models and the rendering system, but also on the quality of the behavior and movement of objects and characters in the scene. A highly detailed character that is not moving naturally could destroy the perceived realism of the environment. Over the last decade, a lot of research has been done to improve character motion. Many new techniques have been developed either based on motion capture [1,2], physics [3] or procedural methods [4]. Also a lot of research has been done

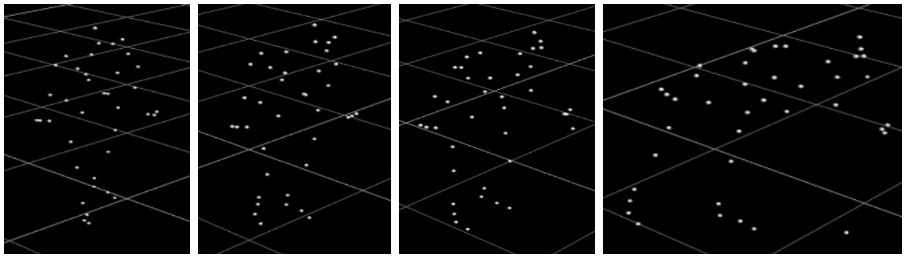

(a) (b) (c) (d)

Fig. 1. Different phases during combined navigation and manipulation actions: (a) navigation, (b) manipulation preparation, (c) reaching, and (d) manipulation

© Springer-Verlag Berlin Heidelberg 2008

on animation from a macroscopic point of view, such as path planning and navigation through an environment with obstacles [5].

In particular, motion synthesis techniques based on example motions are very popular in current games. The inherent realism of performance animation is an attractive advantage of such techniques, as well as the ability to hire actors who can perform a range of motions exactly according to the characteristics of the humans in the game environment. However, recent games contain a vast amount of different characters, all needing different recorded animations, requiring a lot of postprocessing and editing.

A descriptive model of human motion could help to better control virtual characters. However, still a lot is unknown about the details of human movement. This is especially true for more complicated actions, such as combined navigation and manipulation actions: walking to a door and opening it, or walking past a table while picking up an object. Another problem is determining how knowledge obtained about detailed motion can be accurately represented so that it is usable for animating 3D characters. Figure 1 shows an example recording of a combined navigation and manipulation task. In this particular example, the subject was asked to walk to a table and touch the surface at a specific position. As can be seen from the image, there are different phases in this motion, such as: navigation, manipulation preparation, reaching, and the manipulation task itself. Our goal is to find out how these combined tasks are executed and what the different phases in the motion are, in order to establish a *computational model* for human motion based on observed human behavior. Before we discuss the actual experiment and the result, we will first discuss relevant research that is related to our study.

2 Related Work

Motion studies go back for decades. Early studies from Johansson show that people can recognize human motion from point-light display recordings [6,7,8]. Observers of these recordings were rarely able to identify the lights as a human shape when a single frame of the recording was shown. However, in the case of a short motion sequence, they could identify a variety of human motions such as walking and running. Studies done by Cutting and Kozlowski showed that humans are able to identify the gender [9] and even the identity [10] of the recorded subjects by their walk.

Recent research in the field of ergonomics shows that, in certain very specific cases, it is possible to predict the human motion. For example, Wagner et al. [11] shows that it is possible to predict foot placement for specific industrial tasks. Also, the work of Kim et al. [12] shows that if the start and end position of a character are defined, there are various kinematic models (types of movement combinations) that accurately describe a correct transition. Their suggestion to use in-between key-postures to further define the desired motion is already applied in recent motion synthesis techniques, such as the work done by Arikan and Forsyth [13,14].

Nowadays motion capture systems are very popular for recording and analyzing motions, because these systems allow for high-precision, high-framerate 3D recordings. For example, a series of human locomotion studies has recently been done by Arechavaleta et al. [15] using a motion capture system. One of the results of their study is that the trunk generally faces the direction in which the subject is moving. As such, the trunk can be called the *steering wheel* of the human body. The steering wheel concept originates from the mobile robotics research area, where motions are planned with only two velocities: the linear and the angular velocity. By knowing more about how the steering wheel functions for humans, it could be possible to automatically detect when certain tasks are being executed, or to analyse the curvature of the path that humans take toward a goal [16]. We note that a limitation of using motion capture systems is that it does not record all relevant motions. For example, eye movement is not recorded although this has a significant influence on the quantity of movement of the head [12,17].

The previous studies are an indication that motion can indeed be predicted on some level and that perhaps a computational model can be constructed that predicts parts of human motion reasonably well. Although the previous studies already show many valuable insights into human motion, there are few studies that look at more complicated tasks, such as combined navigation and manipulation actions. Additionally, most of the previously discussed studies are not focused on creating a computational model of motion, which is required if one wants to apply the model to virtual character motion. The study presented in this paper is a first step toward establishing a model of combined navigation and manipulation actions by studying human subjects. In this paper, we focus on a particular combination of actions, namely walking toward a door (navigation) and opening it (manipulation). We used a motion capture system to record the subjects in a test environment. This allows us to analyze detailed aspects of human motion.

This paper is organized as follows. In Section 3 we will present the global setup of the experiment and the hypotheses that we will test. Section 4 summarizes the results of the experiments. Finally, we provide some conclusions and we outline our current and future work in Section 5.

3 Experiment Setup

The goal of this experiment is to gain insight on how people approach and open a door. In the experiment, we will examine subjects who open the door with and without the intention of going through it. Also, we will look at the difference between a door that opens inwards versus a door that opens outwards. Since most virtual characters are animated using joints, we do not consider muscle and skin motions in this experiment.

3.1 Participants and Environment

The experiment is done with a group of ten subjects, with equally distributed gender and handedness. We have selected participants of similar age, weight and

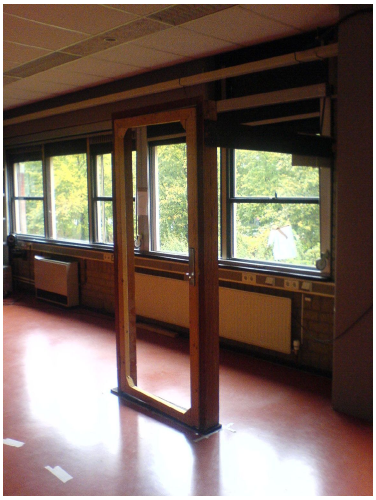


Fig. 2. The door frame in the motion capture lab

height, in order to minimize the possible influence of these variables on the experiment outcome. None of the participants have disabilities or other constraints which could influence their motion. All the data is recorded with a Vicon optical motion capture system[18].

The door used in the experiments is represented by a transparent frame (see Figure 2). The door latch is located at the right side of the door when opening it inwards and on the left side when opening it outwards. We assume in this paper that performing the experiments with the door latch reversed would result in the same data, only mirrored.

3.2 Hypotheses

Before starting the experiments, we formulate a number of hypotheses that we would like to examine. We have grouped these hypotheses into three categories, each of which will be outlined briefly in this section.

The Steering Wheel of the Human Body. In this paper, we define the steering wheel for human motion as the part of the object that determines the direction in which the object is moving. A similar definition is used in research done by Arechavaleta et al. [15]. In order to understand the function of the steering wheel in human motion, we formulate the following hypotheses:

1. While navigating, the trunk serves as the steering wheel for human motion.
2. We expect that gender has no influence on the function of the steering wheel.

The Hand Used to Open the Door. In order to gain insight on the hand that is used to open the door, we will analyze the following hypotheses:

1. In cases where the people have to pass through the door they prefer to use the hand which will not block their way.
2. We expect that gender has no influence on which hand is used to open the door.

The Pose of the Subject When Opening the Door. Especially when planning both navigation and manipulation motions, it is important to know the pose of a character when it starts executing the desired tasks. Navigation motions can be planned so that the character arrives in this pose, and manipulation motions can be planned to start from this pose. We will examine the pose of the subjects in detail, by looking at the relation between the different body parts involved and how this depends on variables such as handedness and gender.

3.3 Experiments

We have performed three different experiments with each of the ten participants. Figure 3 shows the global setup of the experiments. Each experiment was conducted from five different starting positions.

In the first experiment we asked the subjects to walk towards the door and open it as if someone rang the doorbell. The door is positioned in such a way that it opens toward the subject who is opening the door. The door-latch is located on the right side of the door. This experiment consists of five takes corresponding to the five different starting positions.

In the second experiment, the subject has to leave through the door, going either left, right, or straight ahead afterwards. These three goal directions in combination with the five starting positions result in a total of fifteen takes. The door opens inwards.

Finally, the third experiment is a variation on the second where the door opens outwards. This changes the location of the door-latch from right to left

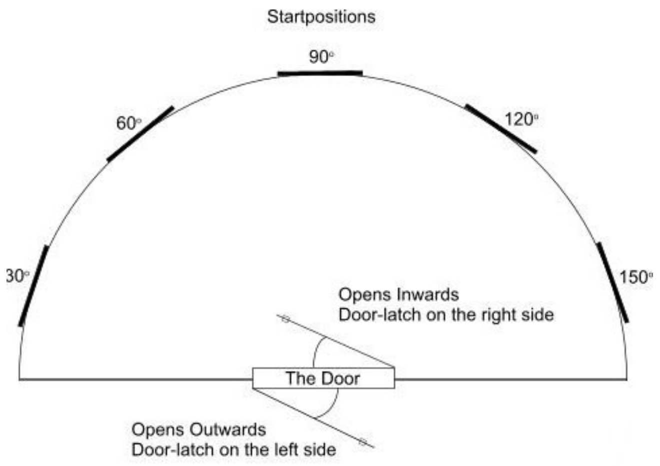


Fig. 3. An overview of the setup of the experiments

side. Again, fifteen takes are required in order to record all the combinations of starting positions and goal directions.

4 Results

We divided this chapter into three sections, corresponding to the three categories of hypotheses formulated in Section 3. First, we are going to look at the steering wheel of human motion. Secondly, we will examine the choice of the hand used to open the door. Finally, we will study the way the subjects stand in front of the door when they open it. Where possible we added ANOVA [19] results.

4.1 The Steering Wheel of the Human Body

Following Arechavaleta's approach, we have looked mainly at the head, the pelvis, and the trunk motion of the recorded subjects. The steering wheel principle applies very well to navigation, as will be shown in the following paragraphs, but as soon as the subjects started the manipulation task, the movement of the different body parts becomes unpredictable. This already happens during the preparation phase of the manipulation. In order to determine the steering wheel of the human body, we will solely look at the motion up until the manipulation preparation phase is entered.

Table 1. Average angles and standard deviation of the pelvis, the trunk and the head with respect to the door

	Average angle	Standard deviation
Pelvis	10,879814275	4,918092469
Trunk	9,364973875	5,678408848
Head	13,883943275	8,270573338

Head. Comparing the average angle between the direction of movement and the direction the head faces shows that the majority of the takes—93 percent—stays within an angle of 30°. As shown in table 1 the average angle error of the head is much higher than the average error for the pelvis and the trunk. Also the standard deviation is remarkably higher.

We have also noticed that the head tends to face the first obstacle the subject has to navigate towards to. As shown in figure 4 this is especially noticeable in the takes where the subjects started from the right side of the door. The average angle error between the direction of movement and the direction the head faces for all the experiments is 13.88°. When the subjects started at an angle of 30° from the door this difference was even bigger: 22.20°. We expected that the difference between the start positions located at the steep angles—30° and 150°—to be small. However, the result of our experiment shows that the startposition located most left from the door at the angle of 150° had a much smaller angle error of 12.23°. Even the start postion at an angle of 60° relative

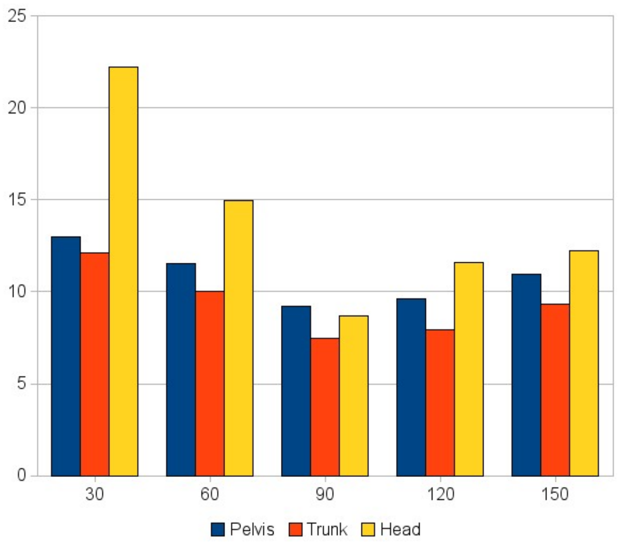


Fig. 4. The distribution of the angular difference between the direction of movement and the direction which the pelvis, trunk, or head are facing per start position

to the door had a larger difference between both directions than the average difference for all the takes.

We suspect that this behavior is related to the location of the door-latch and the direction in which the door opens. In most of the experiments the door-latch was located at the right side of the door, closest to the start position of 30°. When subjects approached the door from that angle they had a shorter path but more importantly they had to *step back* while opening the door. This behavior is already noticeable at the approaching paths. When the subjects have to pass through the door (experiments 2 and 3), they tend to walk to a position next to the door that allows them to open the door and pass through it without having to step back. In this case the subjects did not look at where they are going, but they focus on the door. The looking angle of the head clearly depends on many other factors than path and direction only. Therefore, we agree with Arechavaleta that the head is not a suitable candidate for defining the steering wheel of the human body.

Pelvis. Compared to the head the pelvis follows the direction of movement much better. The average angle error between the direction of movement and the direction the pelvis faces is much less than the angle error of the head joint (see again table1). Looking at these values the pelvis seems a good candidate for the role of steering wheel of the human body. Unfortunately, due the nature of our gait, the direction of the pelvis changes every step, resulting in a periodic sinusoid curve. Although such a curve could be extracted automatically to obtain the correct path and heading, the trunk is a much better candidate, as will be shown in the following paragraph.

Trunk. The trunk, the joint between both shoulders, was proposed by Arechavaleta et al.[15] as the steering wheel of human locomotion. The results of our experiment confirms this statement. Compared to the results from the head and the pelvis, the average angle of the trunk is the smallest (9.37°), although the standard deviation is slightly higher than for the pelvis.

When we compare the results per experiment we see that in all experiments the trunk has the smallest average error angle of all three body parts. Even the minimum of the averages is in all experiments the trunk. Also, the trunk does not tend to face in the direction of the object with which the subject will interact. Neither does the trunk suffer from the periodic rotation that we observed with the pelvis.

4.2 Which Hand Is Used to Open the Door

In this section we will take a closer look at the hand the subjects used to open the door. The general results show that in almost 30% of the cases the subjects chose to open the door with their right hand. In the remaining 70% they used their left hand.

Both gender and handedness show by themselves no significant influence on the hand chosen to open the door. None of the female subjects used their right hand in experiment 1. The male subjects used their right hand in 32% of the takes during experiment 1. The left-handed subjects mostly used their left hand for opening the door. In experiment 1 they used their left hand in 88% of the takes, whereas the right-handed subjects used their left hand in 76% of the takes.

The opening direction proves to have a significant influence on the hand used to open the door with ($P < 0.01$[1]). Experiments 2 and 3 only differ in the direction in which the door opens. Of all subjects more than 91% opened the door with their left hand in experiment 2. In experiment 3 this was just 38%.

The interaction between starting position and the direction in which the door opens is significant ($P = 0.043$). In experiment 2 the subjects used their left hand to open the door in over 90% of the takes no matter from which starting position they started. In experiment 3 the startposition does have a huge influence on the hand used. When the subjects started from the most right starting position, 30° angle to the door, they used their right hand in 80% of the cases. From the starting position at an angle of 60° this was just two third of the subjects. When the subject started from an angle of 120° only half of the subjects used their right hand. At the starting position located at the uttermost left position of the door, at an angle of 150°, a slight gain in right handed use is noticed, resulting in a total of 61%.

A general trend that we notice in these experiments is that the *ergonomy of the motion* seems to preside over the usage of the preferred hand or the starting position relative to the door.

[1] This describes the chance that the relation found between the variables is a random effect.

4.3 The Pose of the Subject When Opening the Door

We will examine the pose of the subject while opening the door in two different ways. First we will look at the position of the subject relative to the door when opening it. Then we will look at the distribution of the body parts (legs, trunk, arms) when the door is opened.

Position Relative to the Door. We studied the position of the subjects and distance to the door while opening it with the data of the first experiment. Our analysis shows that the relation between the starting position and hand chosen to open the door is significant ($P < 0.01$). The starting position located at the most right has an end position that is also located on the right side of the door-latch. The further the starting positions move to the left the more the end positions move to the left. Figure 5 shows the end positions for all starting positions of all subjects in the first experiment.

Gender alone does not appear to have any significant influence on the hand used to open the door. Gender together with the starting positions shows is a significant relation ($P < 0.01$). Comparing the end positions between male and female subjects for all the takes shows a larger spread with the male subjects. The position of the male subjects ranges from 40cm to the right of the door-latch to 34cm to the left. For the female subjects this spread was smaller and ranged from 18cm to the right to 34cm to the left.

The relation between the starting position and handedness is significant ($P = 0.037$). Left-handed subjects seem stand more left to the door than the right-handed subjects. From the starting position at 30°, the difference between left

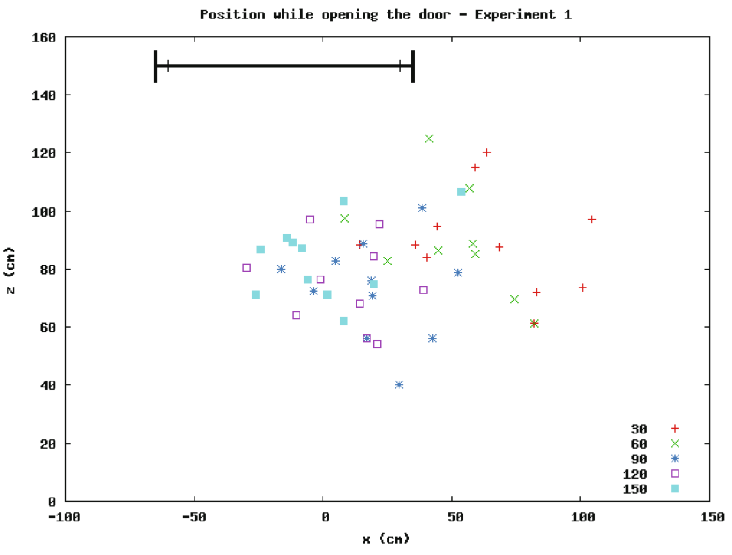


Fig. 5. The position relative to the door while opening it

and right-handed subjects is more than 20cm. The closer the subjects start from startpositions at the left side the smaller the difference becomes.

Distance Relative to the Door. We define the total distance of the subject from the door as the distance from the ankle of the rear leg to the hand which opens the door (see also figure 6). The body parts analyzed are the rear leg, the upper body and the reaching arm. Our analysis shows that on average most of the distance is covered by the arms (more than 70% of the total distance). The legs cover almost 20% and the remaining 10% is covered by the upper body.

The relation between the distance covered by the reaching arm and the position from which the subjects started is significant ($P < 0.01$). When the subjects approached the door from the right side, the starting positions at an angle of 30° and 60° they reach less with their arm than when they approach the door from upfront or from the left side. When the subjects approached the door from the right side, they reach less with their arm than when they approach the door from upfront or the left side. The recordings show that the subjects place themselves at a comfortable position relative to the door. This seems to indicate that comfort of a motion is very important, even it the most comfortable motion requires extra energy.

There is a significant relation between gender and the distance to the door for the upper body ($P = 0.049$). In a number of cases subjects actually leaned backwards while opening the door. The most remarkable detail is that this only happend to female subjects. Out of the 50 takes the female subjects had 15 cases in which the pelvis was closer to the door than the torso.

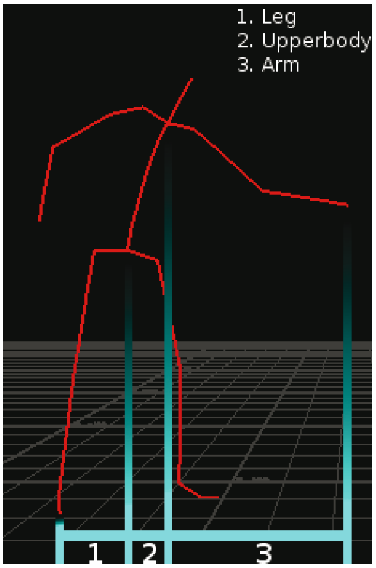


Fig. 6. The distribution of distance per body part

5 Conclusion and Future Work

In this paper we have discussed the results of a motion analysis experiment where subjects have to walk to a door and open it. We have seen significant relations between variables that give us some insight in how people combine navigation and manipulation actions. We have observed that the trunk can be described as the steering wheel of the human body while navigating. This may form a useful tool to automatically identify when a manipulation action is started. Another more global observation that follows from our experiments is that the *comfort* of the motion is a very important factor: people tend to choose the motion that is the most comfortable, even if this means walking a longer path.

Although this experiment provides us with some clues on how to construct a computational motion of human motion, it is only a starting point. Our goal is to analyse in more detail how the different phases during these combined navigation and manipulation actions occur. We are currently performing a number of experiments that should give a clearer image of this. We hope that through these experiments, we will be able to contruct a more complete computational model of human motion and behaviour.

Acknowledgements

This research has been supported by the GATE project, funded by the Netherlands Organization for Scientific Research (NWO) and the Netherlands ICT Research and Innovation Authority (ICT Regie).

References

1. Kovar, L., Gleicher, M., Pighin, F.: Motion graphs. In: Proc. SIGGRAPH (2002)
2. Egges, A., Molet, T., Magnenat-Thalmann, N.: Personalised real-time idle motion synthesis. In: Pacific Graphics 2004, pp. 121–130 (2004)
3. Badler, N.I., Metaxas, D., Kokkevis, E.: User-controlled physics-based animation for articulated figures. In: Proceedings of Computer Animation, pp. 16–26 (1996)
4. Cassell, J., Vilhjálmsson, H., Bickmore, T.: BEAT: the Behavior Expression Animation Toolkit. In: Proceedings of SIGGRAPH, pp. 477–486 (2001)
5. Nieuwenhuisen, D., Kamphuis, A., Overmars, M.H.: High quality navigation in computer games. Science of Computer Programming 67, 91–104 (2007)
6. Johansson, G.: Visual perception of biological motion and a model for its analysis. Perception & Phychophysics 14(2), 201–211 (1973)
7. Johansson, G.: Visual motion perception. Scientific American 232(6), 76–88 (1975)
8. Johansson, G.: Spatio-temporal differentiation and integration in visual motion perception. an experimental and theoretical analysis of calculus-like functions in visual data processing. Psychological Research 38(4), 379–393 (1976)
9. Kozlowski, L.T., Cutting, J.E.: Recognizing the sex of a walker from a dynamic point-light display. Perception & Psychophysics 21(6), 575–580 (1977)
10. Cutting, J.E., Kozlowski, L.T.: Recognizing friends by their walk: Gait perception without familiarity cues. Bulletin of the Psychonomic Society 9(5), 353–356 (1977)

11. Wagner, D.W., Reed, M.P., Chaffin, D.B.: Predicting foot positions for manual materials handling tasks. In: Digital Human Modeling for Design and Engineering Symposium, Iowa (June 2005)
12. Kim, K.H., Martin, B.J., Gillespie, R.B.: Posture and motion prediction: Perspectives for unconstrained head movements. In: Digital Human Modeling for Design and Engineering Conference, Lyon (July 2006)
13. Arikan, O., Forsyth, D.A., O'Brien, J.F.: Motion synthesis from annotations. In: Proceedings SIGGRAPH, San Diego, California, pp. 402–408. ACM, New York (2003)
14. Arikan, O., Forsyth, D.A.: Interactive motion generation from examples. ACM Trans. Graph. 21(3), 483–490 (2002)
15. Arechavaleta, G., Laumond, J.-P., Hicheur, H., Berthoz, A.: The nonholonomic nature of human locomotion: a modeling study. In: International Conference on Biomedical Robotics and Biomechatronics, Pisa, Italy, pp. 158–163 (2006)
16. Arechavaleta, G., Laumond, J.-P., Hicheur, H., Berthoz, A.: Optimizing principles underlying the shape of trajectories in goal oriented locomotion for humans. In: International Conference on Humanoid Robots, pp. 131–136. IEEE, Los Alamitos (2006)
17. Shimazaki, C., Uemura, T., Arai, Y.: Eye-head coordination during lateral gaze in normal subjects. Acta Otolaryngol 90(3–4), 191–198 (1980)
18. Vicon website (Accessed, April 2008), `http://www.vicon.com`
19. Ferguson, G.A., Takane, Y.: Statistical Analysis in Psychology and Education, 6th edn. McGraw-Hill Ryerson Limited, Montreal (2005)

Watch Out! A Framework for Evaluating Steering Behaviors

Shawn Singh, Mishali Naik, Mubbasir Kapadia, Petros Faloutsos, and Glenn Reinman

University of California, Los Angeles

Abstract. Interactive virtual worlds feature dynamic characters that must navigate through a variety of landscapes populated with various obstacles and other agents. The process of navigating to a desired location within a dynamic environment is the problem of *steering*. While there are many approaches to steering, to our knowledge there is no standard way of evaluating and comparing the quality of such solutions. To address this, we propose a diverse set of benchmarks and a flexible method of evaluation that can be used to compare different steering algorithms. We discuss the challenges and criteria for objectively evaluating steering behaviors and describe the metrics and scoring method used in our benchmark evaluation. We hope that, with constructive feedback from the community, our framework will eventually evolve into a standard and comprehensive approach to debug, compare and provide an overall assessment of the effectiveness of steering algorithms.

1 Introduction

A fundamental requirement of nearly all agents in virtual worlds is *steering*: the ability of an agent to navigate to a goal destination, through an environment that includes static obstacles, such as buildings, and dynamic obstacles, such as other virtual characters. Steering is a challenging problem for autonomous agents. In reality, steering is a result of a complex process: An agent makes steering decisions based on sensory information, predictions of the motion of dynamic obstacles, social etiquette, personal experience, situation specific parameters, cognitive goals and desires. Even within the simplified environment of a virtual world, the state space of the steering problem is too large to allow for trivial solutions, such as pre-computed state-action tables.

There is a significant amount of research that tries to address the steering problem. Current solutions seem to address only a subset of the problem's challenges. For example, particle-based approaches are well suited for macroscopic crowd behaviors, while agent-based approaches work better for the local interaction of a small group of agents.

Given the importance of steering in modern applications and the growing number of steering algorithms, it is important and timely to ask the question, *how can we compare different steering approaches?* To our knowledge this paper makes the first attempt to answer this fundamental question.

© Springer-Verlag Berlin Heidelberg 2008

We propose a novel benchmarking tool for comparing steering behaviors. Our framework is based on two components that form the main contributions of this paper:

- *A diverse suite of steering benchmarks*: We propose a forward-looking set of scenarios that capture the broad range of situations a steering algorithm may encounter in practical applications.
- *A method and metrics of evaluation*: We propose a set of metrics that can be used to evaluate the behavior of steering algorithms. We also propose a method of scoring the behavior, so that two steering algorithms can be compared. We call this process *benchmark evaluation.*

1.1 Criteria for Effective Evaluation

The main challenge in designing steering benchmarks is how to evaluate a steering algorithm *objectively.* Determining the quality of the results of a steering algorithm depends on many factors, many of which are situation-specific and/or depend on cognitive decisions by the agents. For example, an agent may decide to push through a crowd or politely go around the crowd, depending on the agent's situation and personality. To remain objective even with seemingly ad hoc constraints, we propose that steering evaluation should satisfy the following criteria:

1. The test cases should be representative of a broad range of situations that are common in real-world scenarios.
2. The evaluation should be blind to the specifics of the steering algorithm.
3. The evaluation should be customizable to allow a user to test for certain expected behaviors over others.

To meet these criteria, we separate the concepts of providing test cases and evaluating a steering result. This allows users to specialize test cases to their needs, and our benchmark evaluation would still apply to the custom test cases. Our evaluation uses metrics computed from the position, direction, and the goal that the agent is trying to reach. The interpretation of these metrics is surprisingly meaningful, yet at the same time, they do not require any knowledge of the algorithm that produced the steering result.

Of course, no set of benchmarks and associated metrics could cover and evaluate every possible steering situation that can happen in the real world – and we do not claim that ours does. We cannot necessarily tell how well one algorithm does in absolute terms. Instead, our framework gives a good indication of the pros and cons of any steering algorithm and to effectively *compare* different approaches.

2 Related Work

Benchmarking is a crucial process in many fields – ranging from business management to software performance. Benchmark suites have been developed for a variety of purposes related for graphics and multimedia, including hardware [1], global illumination [2], animation for ray tracing [3], general-purpose architecture [4], and many more. In these fields, benchmarks have clear metrics for comparison: performance in seconds, signal-to-noise ratio, power consumption, area, monetary price, etc. Steering behaviors and other aspects of artificial intelligence do not have a clear objective metric – instead, much of the evaluation is inherently subjective.

Efficiency Metrics. Some works have proposed metrics that evaluate "believability" or "natural behavior." One major approach is to follow the rule that natural behaviors are usually efficient (e.g. [5]). This approach has been used effectively for animation, e.g. [6,7]. Our work applies this same principle to the evaluation of steering behaviors, while keeping the user in control of the final evaluation process.

Approaches to Steering. Most steering behaviors can be classified into two major categories: Dynamics-based and agent-based. Dynamics based models represent the environment with potential fields or flow maps, often treating each agent as a particle (e.g., [8,9,10]). Rule-based models use heavy branching to determine the exact scenario being described, and performs an action associated with that scenario (e.g., [11,12,13,14]). A problem is that such papers only have enough space to showcase their novel features, and it is difficult to assess how effective a steering algorithm will be on fundamental, common scenarios that were not the focus of the paper. A benchmark suite can illuminate these results in a compact form that future papers could very easily include.

3 Benchmark Suite

Our framework consists of two major parts: (1) a diverse set of test cases, and (2) a benchmark evaluation utility that uses several metrics to score a steering algorithm. In this section, we describe the suite of test cases.

Overview. Our current framework contains 36 scenarios. Several of the scenarios have many variations, resulting in a total of 50 test cases. The test cases can be classified in five major categories: (1) simple validation scenarios, (2) basic one-on-one interactions, (3) agent interactions including obstacles, (4) group interactions, and (5) large-scale scenarios. Many test cases can be classified in multiple categories – this classification is only for presentation, and does not limit the test cases or evaluation process in any way. These five categories are detailed in Section 3.1.

Each test case specifies a number of agents and static objects. Static objects are specified by a bounding box. Agents are specified by their size, initial position, initial direction, desired speed, and target location(s). Additionally, each test case specifies which agents should be examined during benchmark evaluation, and a set of weights that configure the evaluation process, discussed in Section 5. Dynamic objects can be specified as agents in a customized scenario, however, our current suite does not test agent's steering behaviors with dynamic objects because the expected behaviors in such cases are application-specific.

Test Case Design Choices. The main challenge of designing the benchmark suite is to cover the range of possible scenarios without having an inordinate number of test cases. With this in mind, we choose our test-cases to be common, frequently appearing scenarios, but with challenging, worst-case parameters. For example, most of the static obstacles have sharp corners which are generally more difficult to handle than smooth ones.

In our experience, an agent in a typical urban environment faces the following situations:

1. *Walking alone.* Most of the time, an agent interacts with only 2-4 nearby agents and a few obstacles at a time.
2. *Walking as part of a small group.* Often the agent moves in the same direction with 2-6 other agents and encounters other groups and obstacles.
3. *Walking as part of a crowd.* Groups of hundreds of agents tend to form less frequently, and only in specific situations, for example when a large number of agents exit or enter a building at the same time.

Furthermore, some of the situations involving many agents, can often be viewed as a sequence of smaller problems. For example, a lone agent that tries to go through a dense oncoming crowd of hundreds of agents, may only consider 4-5 people at a time. Based on these observations, we choose test cases that reflect these smaller problems which can represent a large number of composite real-world scenarios.

It is fair to ask the question, what is the effect of the specific number of agents in the large scale examples? We have selected default values for these parameters based on what we think are average cases in the real world. We have no reason to believe that the specific number of agents in the large scale examples is crucial. If an algorithm can handle a bottleneck example with exactly 500 agents, it should be able to handle around 500 as well. In any case, we would like to remind the reader that our goal is to provide an estimate of an algorithms performance, not a proof of its robustness or correctness.

Customizing the Suite. It is clearly not possible to cover every conceivable situation in our test cases. Therefore, we have designed our benchmarking approach to be flexible and customizable, so that users can quickly focus on the details of interest to their algorithm.

The user can easily create custom scenarios to use with our existing benchmark evaluation. This allows our benchmark evaluation process to be useful even for applications that cannot use our provided test cases. For example, steering behaviors found in a sports game will have unique steering scenarios that should be evaluated with unique criteria.

The initial conditions and parameters can also be re-defined by the user. In its current form, our test cases are intended to roughly approximate typical humans: agents have a diameter of 1 meter (roughly the distance from elbow to elbow of an average human) and an average walking speed of 1.3 meters per second [15]. The user can use slightly modified initial conditions to simulate cars, bicyclists, or any dynamic objects.

3.1 Description of the Scenarios

Here we describe the scenarios used in the current version of our benchmark suite. These scenarios can be seen in Figure 1. Many of these test cases are very challenging and forward-looking, and we expect that initially very few algorithms, if any, will be able to successfully handle all scenarios gracefully. Scenarios annotated with an asterisk (*) are in our opinion more difficult scenarios.

Simple Validation Scenarios. These scenarios are designed to test very basic, fundamental abilities of a steering agent. While such behaviors are trivial, it is still important for an algorithm to successfully handle these cases.

- *Simple:* one agent steering towards a target located to the left, right, or behind.
- *Simple-obstacle:* one agent steering towards a target located behind an obstacle.
- *Simple-wall:* two agents steering around a wall to reach a target.
- *Curves:* one agent steering through an S-curve to reach the target.

Basic One-on-one Interactions. These scenarios test the ability of agents to steer around each other. In this category, the emphasis is on natural interactions without other threats to distract the agents.

- *Oncoming:* two agents traveling in opposite directions towards a head-on collision.
- *Crossing:* two agents crossing paths at various angles.
- *Oncoming-trick:* two oncoming agents that will not collide because their targets are closer.
- *Crossing-trick:* two crossing agents that will not collide because their targets are closer.
- *Similar-direction:* two agents with slightly different goals traveling in a similar direction.

Agent-agent Interactions Including Obstacles. These scenarios test the ability of an agent to navigate around static objects while interacting with other agents.

- *Oncoming-obstacle:* two oncoming agents, with an additional obstacle in the way.
- *Crossing-obstacle:* two crossing agents, with an additional obstacle in the way.
- *Surprise:* two agents that do not see each other until the last minute because of large obstacles.
- *Squeeze:* two oncoming agents walking through a narrow hallway.
- *Doorway-one-way:* two agents enter a doorway from the same side.
- **Doorway-two-way:* two oncoming agents walk through a doorway from opposite sides.
- *Overtake:* one agent is expected to overtake the other agent in a hallway.
- **Overtake-obstacle:* one agent is expected to overtake the other agent, with obstacles in the hallway.

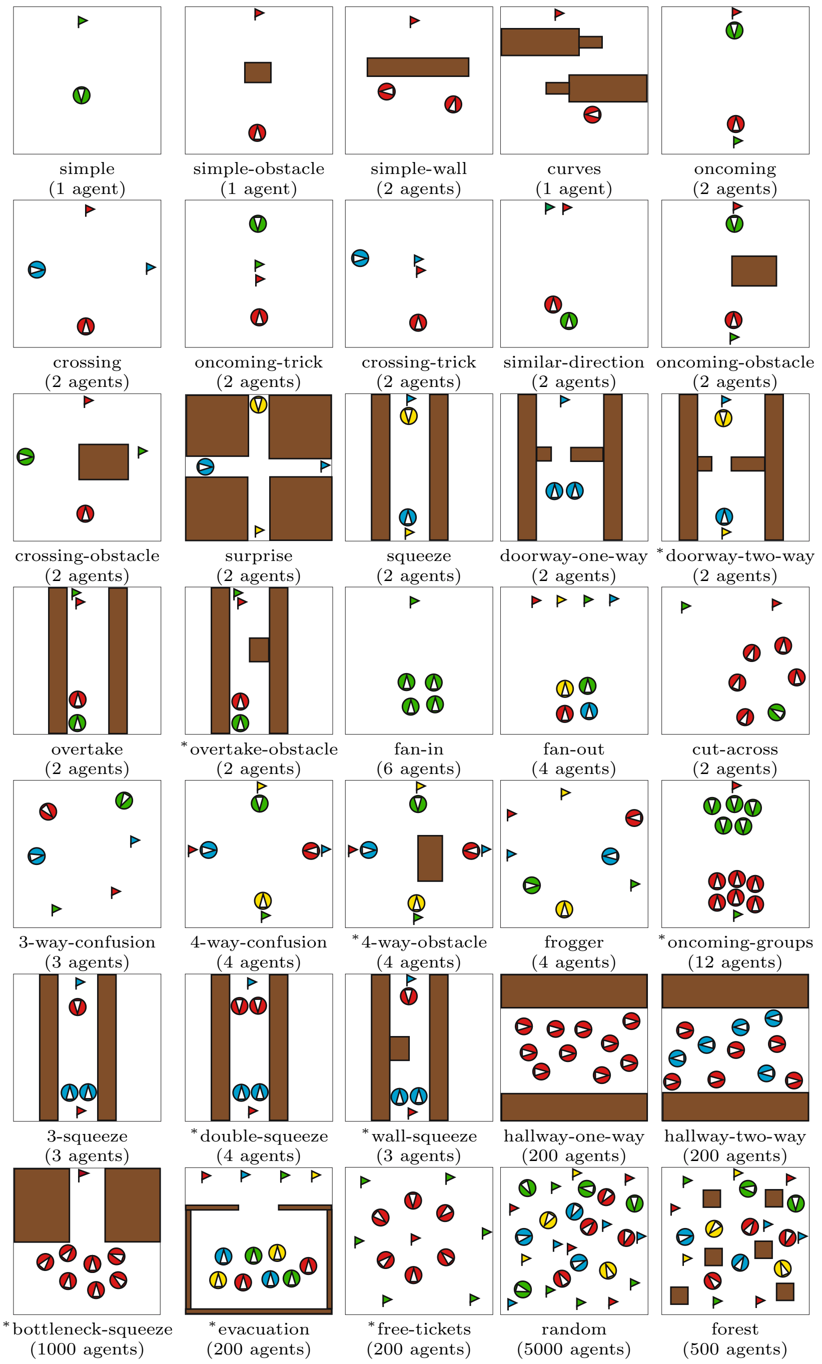


Fig. 1. Approximate depiction of each scenario in the current version of our benchmark suite. The number in parentheses indicates how many agents are specified in the scenario, and asterisks indicate more difficult scenarios.

Group Interactions. Group interactions are composed of several agents and static objects, intended to test an algorithm's ability to handle a variety of common situations.

- *Fan-in:* a small group of agents aiming for the same target. This tests how agents cooperate while contending for the same space.
- *Fan-out:* a small group of agents aiming for slightly separated targets. This tests whether agents will unnaturally stick to the crowd when their goal is in a different direction.
- *Cut-across:* one agent cutting across a small group.
- *3-way-confusion:* three agents traveling in different directions, meeting at nearly the same time.
- *4-way-confusion:* four agents traveling in four opposing directions, meeting at nearly the same time.
- **4-way-obstacle:* four agents crossing paths with a static object in the way.
- *Frogger:* one agent encounters many perpendicular crossing agents.
- **Oncoming-groups:* a small group of agents encounters another small group of agents traveling in opposite direction.
- *3-squeeze:* two agents facing the same direction encounter an oncoming agent in a narrow hallway.
- **Double-squeeze:* two agents facing the same direction encounter two oncoming agents in a narrow hallway.
- **Wall-squeeze:* two agents facing the same direction encounter an oncoming agent in a narrow hallway with an obstacle.

Large Scale Scenarios. These scenarios are designed to stress-test the ability of an algorithm to handle macroscopic situations, and to scale to a large number of agents.

- *Hallway-one-way:* many agents traveling in the same direction through a hallway.
- *Hallway-two-way:* many agents traveling in either direction through a hallway. Agents should form lanes.
- **Bottleneck-squeeze:* all agents begin on one side of the arena, and must enter and traverse a hallway to reach the target. Note that hard corners at the bottleneck are much more challenging than rounded corners.
- **Evacuation:* all agents must exit a crowded room that has only one exit.
- **Free-tickets:* all agents are aiming for the same target in the middle of the arena, and have a random secondary goal once the middle target is reached. This scenario is particularly difficult because agents that reach the middle goal must then turn to face a dense oncoming crowd. This scenario requires both natural individuals and natural crowds simultaneously.
- *Random:* each agent is placed randomly in the arena and has an individual random target. Here, stress is placed on handling a large number of agents. Our default test case specifies 5000 agents for this scenario.
- *Forest:* each agent is placed randomly in an arena filled with small obstacles.
- *Urban:* each agent is placed randomly in an arena filled with building-sized obstacles.

4 Metrics of Evaluation

Given the suite of benchmarks, the next question is how to evaluate the result of a steering algorithm. We propose that a set of meaningful metrics can be measured directly from the output of a steering algorithm, without any knowledge about the algorithm itself. This section describes the metrics that we compute.

Our benchmark evaluation works as follows. First, the user records the position, direction, and goal target of every agent for every frame. Second, our evaluation tool will read this recorded information to compute a variety of statistics about each agent's behavior. Users can see these detailed statistics or automatically computed scores based on three primary metrics.

Primary Metric 1: Number of Collisions. The first primary metric is the number of collisions that occur for a given agent. In most cases, fewer collisions indicates more realistic steering behavior. One notable exception is the *Surprise* scenario, where it would be natural for two agents to collide because they do not see each other soon enough. Our evaluation tool computes the number of unique collisions that occur as well as the total number of frames that each agent spent in a collision with other objects.

Primary Metrics 2 and 3: Time and Effort Efficiency. The second and third primary metrics measure two forms of efficiency. These, and most of the detailed metrics as well, are based on the idea that *efficient* behaviors are very often natural behaviors. Section 2 describes many references that use the same principle.

Time efficiency measures how quickly the agent is able to reach its goal destinations. Of course, the quicker the agent reaches its goal, the more time efficient the agent is. Our evaluation tool measures time efficiency as the total time (in seconds) that an agent spent to reach its goal.

Effort efficiency measures how much effort an agent spent to reach its goal destinations. The less effort an agent spent, the more effort efficient the agent is. Our evaluation tool measures effort efficiency as the integral (sum total) of the magnitude of acceleration that an agent used to reach its goal. Note that acceleration includes changes in speed and changes in direction, and its magnitude is always positive.

The combined interpretation of time and effort efficiency provides insight into a spectrum of behaviors. Some agents may desire to reach their destinations quickly, willing to spend more effort. Other agents may desire to save effort and slowly, politely progress towards their goals.

Detailed Metrics. The rest of the metrics measure variations of: (1) speed, (2) turning rate (angular velocity), and (3) change in speed. Specifically, for each of these three, we measure the total (integral), max instantaneous, average, and max/min over a sliding window. The sliding window in our current implementation is an integral over an interval of 20 frames. We compute a new window integral every next frame, and finally store the max/min of these values.

These detailed metrics are interesting to examine when a user knows the expected behavior of a scenario. A user will usually be able correlate one or two of the metrics with a clear indication of unnatural behavior in the scenario. For example, if a character is expected to go straight towards its goal with very little turning, then the "integral of turning rate," which describes the total amount of turning, should be close to zero. Similarly, if an agent is expected to have one abrupt turn in the scenario, the "integral of turning rate in a window interval" should be somewhat large, while the "total average turning rate" should be small.

5 Benchmark Scoring

After computing the metrics described in the previous section, the final task is to compute a meaningful score that represents the quality of the steering behavior. A score can be computed for (1) a single agent in a test case, (2) all agents in a test case, or (3) across all test cases.

A Note About Benchmark Scores. It is important to note that scoring is not intended to be a proof of an algorithm's effectiveness. Instead, the purpose of scoring is to create *a simple number* that allows for a quick, intuitive estimate of evaluation, especially when comparing two approaches. To do a more rigorous analysis of the pros and cons of an algorithm, users need to manually examine the detailed metrics, and perhaps even tailor their own test cases.

Scoring One Agent in One Test Case. An agent's score can be computed by combining the three major metrics described in Section 4: number of collisions, time efficiency, and effort efficiency. The user describes their relative importance as numerical weights, specified in the test case. Each agent in the test case has its own unique set of weights.

The score is simply a weighted sum of the metrics:

$$S_i = w_c C + w_t T + w_e E, \tag{1}$$

where S_i is the score of the i^{th} agent, w_c, w_t, and w_e are the user-specified weights, C is the number of collisions, T is time efficiency, and E is effort efficiency.

Scoring all Agents in One Test Case, and Across all Test Cases. To evaluate all agents in one test case, we simply compute the average over n agent scores:

$$S_A = \sum_{i=0}^{n} S_i. \tag{2}$$

Because each agent can have its own set of weights, the user can easily control the relative importance of agents in the scoring process.

Finally, to score an algorithm across all test cases, we can compute a sum total of the scores from each test case.

$$S_m = \sum_{A} S_A. \tag{3}$$

6 Conclusion

In this paper we present a framework for evaluating steering behaviors. The framework includes a diverse suite of test cases and an objective method of evaluation. The framework covers a broad range of common scenarios, is blind to the specifics of the steering algorithm, and is extensible and customizable. In future work, we plan to continue growing the suite of test cases, make the framework available online, and demonstrate its effectiveness on recordings of real humans steering through our test cases. We envision that this framework can grow into a standard for steering evaluation.

References

1. Futuremark: 3DMark (2008), `http://www.futuremark.com`
2. RealStorm: RealStorm Global Illumination Benchmark (2008), `http://www.realtimeraytrace.de/`
3. Lext, J., Assarsson, U., Moller, T.: A benchmark for animated ray tracing. IEEE Computer Graphics and Applications 21(2), 22–31 (2001)
4. Henning, J.L.: Spec cpu2006 benchmark descriptions. SIGARCH Comput. Archit. News 34, 1–17 (2006)
5. Reitsma, P.S.A., Pollard, N.S.: Evaluating motion graphs for character animation. ACM Trans. Graph. 26(4), 18 (2007)
6. Wu, J.c., Popović, Z.: Realistic modeling of bird flight animations. ACM Trans. Graph. 22(3), 888–895 (2003)
7. Tu, X., Terzopoulos, D.: Artificial fishes: physics, locomotion, perception, behavior. In: SIGGRAPH 1994: Proceedings of the 21st annual conference on Computer graphics and interactive techniques, pp. 43–50. ACM Press, New York (1994)
8. Brogan, D.C., Hodgins, J.K.: Group behaviors for systems with significant dynamics. Auton. Robots 4(1), 137–153 (1997)
9. Goldenstein, S., et al.: Scalable nonlinear dynamical systems for agent steering and crowd simulation. Computers and Graphics 25(6), 983–998 (2001)
10. Treuille, A., Cooper, S., Popović, Z.: Continuum crowds. In: SIGGRAPH 2006: ACM SIGGRAPH 2006 Papers, pp. 1160–1168. ACM, New York (2006)
11. Paris, S., Pettré, J., Donikian, S.: Pedestrian reactive navigation for crowd simulation: a predictive approach. In: EUROGRAPHICS 2007, vol. 26, pp. 665–674 (2007)
12. Lamarche, F., Donikian, S.: Crowd of virtual humans: a new approach for real time navigation in complex and structured environments. Computer Graphics Forum 23(10), 509–518 (2004)
13. Loscos, C., Marchal, D., Meyer, A.: Intuitive crowd behaviour in dense urban environments using local laws. In: TPCG 2003: Proceedings of the Theory and Practice of Computer Graphics 2003, p. 122. IEEE Computer Society, Washington (2003)
14. Shao, W., Terzopoulos, D.: Autonomous pedestrians. In: SCA 2005: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation, pp. 19–28. ACM, New York (2005)
15. Knoblauch, R.L., Pietrucha, M.T., Nitzburg, M.: Field studies of pedestrian walking speed and start-up time. Transportation Research Record 1538, 27–38 (1996)

Whole-Body Locomotion, Manipulation and Reaching for Humanoids*

Eiichi Yoshida[1], Jean-Paul Laumond[2], Claudia Esteves[3], Oussama Kanoun[2], Takeshi Sakaguchi[1], and Kazuhito Yokoi[1]

[1] AIST/ISRI-CNRS/STIC Joint French-Japanese Robotics Laboratory (JRL), National Institute of Advanced Industrial Science and Technology (AIST), 1-1-1 Umezono, Tsukuba, 305-8568 Japan
e.yoshida@aist.go.jp

[2] JRL, LAAS-CNRS, University of Toulouse, 7, av. du Colonel Roche, 31077 Toulouse, France
jpl@laas.fr

[3] Facultad de Matematicas - Universidad de Guanajuato, Mexico
cesteves@cimat.mx

Abstract. This paper deals with motion planning and dynamic control for humanoid robots. The first part addresses simultaneous locomotion and manipulation planning while the second part deals with reaching tasks. The validity of the proposed methods is verified by experiments using humanoid platform HRP-2.

1 Introduction

Humanoid robots are expected to perform complicated tasks thanks to their high mobility and many degrees of freedom including legs and arms. Their anthropomorphic configuration gives another advantage that they can easily adapt to machines or environments designed for humans. Recent progress in hardware accelerates diverse research in humanoid robots. Various types of tasks have been performed: manipulation [3,4,5], navigation in dynamic environments [6,7], or serving tasks [8,9]. One of the key issues to fully exploit the capacity of humanoid robots is to develop a methodology that enables them to explore and execute various dynamic tasks, requiring dynamic and smooth whole-body motion including collision avoidance and locomotion, like an object carrying task.

In the field of motion planning, recent advancement in probabilistic methods has greatly improved the three-dimensional (3D) motion planning for mechanism involving complicated geometry and many degrees of freedom (e.g. [26]). However, most of those methods are based on geometric and kinematic planning in configuration space whereas dynamic control is required for humanoid motion planning in workspace to execute tasks while keeping balance.

Concerning control issues of humanoid robots, powerful controllers have been developed to generate whole-body dynamic motion in a reactive manner (e.g.,

* This paper is a compilated version of papers presented in IEEE IROS 2006 [1] and IEEE Humanoids 2006 [2].

© Springer-Verlag Berlin Heidelberg 2008

[13]). As for locomotion, stable motion pattern can be generated efficiently thanks to the progress in biped walking control theory, mainly based on ZMP (zero moment point [10]) control (e.g., [11]). Planning of 3D humanoid motion for tasks in complex environments has to benefit from these two domains.

In the first part of this paper, we propose a two-stage planning framework based on the geometrical and kinematic planning technique whose output is validated by dynamic motion pattern generator. The second part addresses the following problem: how to recover a positioning error of the humanoid robot body when performing manipulation tasks such as reaching or grasping?

2 Manipulating While Walking

Humanoid motion planning is becoming a hot topic since it faces complexity of planning and dynamic control at the same time. Kuffner et al. proposed various types of humanoid motion planner [16,17,18,19] such as balancing, footstep planning and navigation displacing movable obstacles. Locomotion planning for humanoid robots to pass through narrow spaces by changing the locomotion modes has been investigated in [20,21]. Okada et al. addressed motion planning for collision-free whole-body posture control [22] by dividing the robot into movable, fixed and free limbs using RRT planner. They have also showed task-oriented motion planning [23]. Yoshida proposed humanoid motion planning based on multi-level DOF exploitation [24]. Sentis et al. developed a hierarchical controller that synthesizes whole-body motion based on prioritized behavioral primitives including postures and other tasks in a reactive manner [13].

In the domain of computer graphics, motion editing is an active area of research. Gleicher classified various constraint-based methods that take account of spatial and temporal constraints, which often corresponds to the problems of inverse kinematics and filtering respectively [15]. Especially for graphic animation of digital actors, recent development in randomized motion planning is now actively investigated [25,26].

2.1 Two-Stage Planning Method

In this section we summarize the two-stage planning method we have proposed in [14]. The lower part of the robot body is approximated by a bounding box. Then, at the first stage, the motion planner computes a collision-free walking path as well as a collision-free path for the upper body. In the second stage, these outputs are given as inputs to the dynamic pattern generator [11] of humanoid robots. The dynamic pattern generator transforms the paths into a dynamically executable motion described by waist position and orientation and joint angles of whole body at sampling time of 5[ms] by taking account of dynamic balance based on ZMP. However, the generated dynamic motion often differs from the geometrically and kinematically planned path, which may cause unpredicted collision. Then the planner goes back to the first stage to "reshape" the previous path based on randomized method to avoid possible collision. If no solution is found, then a new walking plan is searched. In the next section we address the motion reshaping issues.

2.2 Smooth Motion Reshaping

A collision-free path issued from the first motion planning stage, will not always result in a collision-free trajectory after dynamic pattern generation is performed. For instance, unexpected collisions might arise by the influence of the weight of the carried object into the robot's dynamics. If the variation of the motion is small enough, those collisions will be with the humanoid's upper body or the carried object. In such a case, we can assume that local reshaping of the trajectory will suffice to avoid the obstacles without replanning the whole nominal trajectory.

When a collision is found, a new random collision-free configuration near the colliding one is first generated, and then an inverse kinematics (IK) solver is applied to ensure the end-effector's geometric constraints (see [14] for details). Although collision-free motions can be generated at that stage, lack of smoothness in velocity profile might cause instability or unnecessary oscillation when it is executed by the humanoid robot. Then we propose a reshaping method that accounts for the smoothness of the motion when avoiding the obstacles.

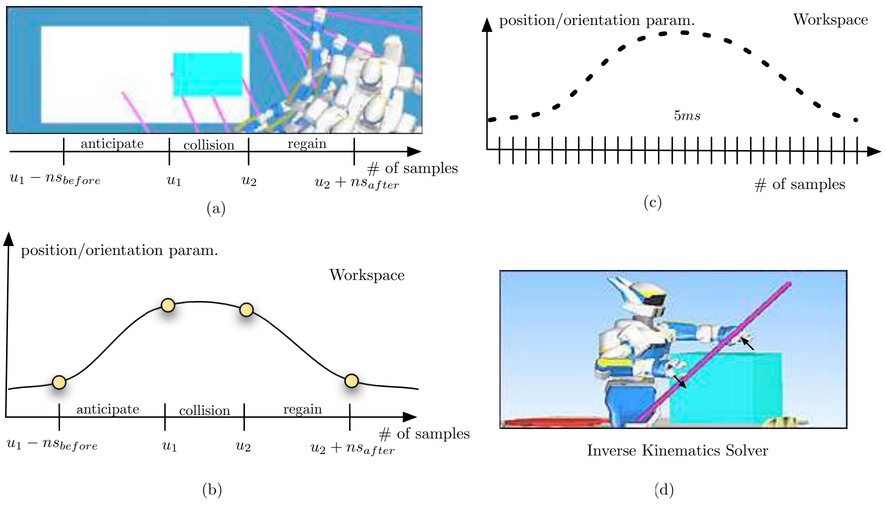
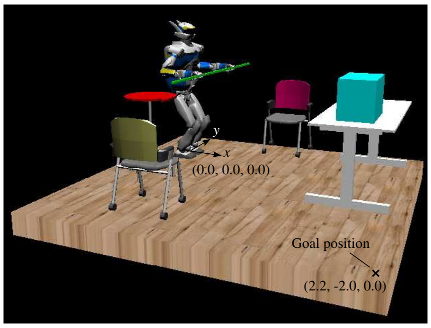


Fig. 1. (a) A top view of the volume swept by the robot avoiding a box on the table. Collisions occur between the bar and the box. The reshaping limits are set by identifying the anticipating, colliding and regaining times. (b) Smooth motion is specified in the task space by interpolating the bar's configuration at key frames. (c) The bar's motion is resampled at 5[ms] to replace its original motion. (d) New constraints are enforced by using a whole-body IK solver.

The reshaping procedure is performed in two steps illustrated in Fig. 1 (see the real situation in Figure 2):

1. A smooth trajectory to be followed by the end-effector is specified in the task space and resampled at each sampling time (5[ms]) to enforce temporal constraints (Fig. 1(a)-(c)).
2. An inverse kinematics solver is used to attain the specified end-effector's motions enforcing geometric constraints (Fig. 1(d)).

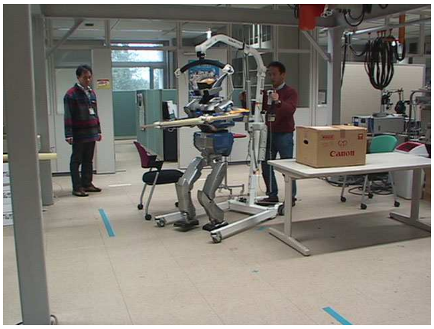


(a) Initial configuration

(b) Start walking

(c) Start lifting the bar

(d) The bar passes above the obstacle


(e) Lowering the bar after avoiding collision

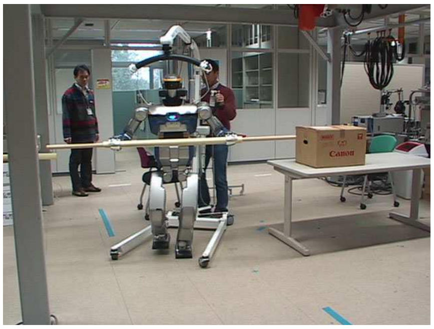

(f) Going to final position

Fig. 2. Experiment of planned bar-carrying task

We account for motion continuity at both steps (see [14] for details).

We have conducted experiments of the proposed humanoid motion planner using simulator OpenHRP [28] and hardware humanoid platform HRP-2 [29]. HRP-2 has 30 degrees of freedom with 1.54[m] in height and 58[kg] in weight. This robot has two chest joints for pitch and yaw rotation, which extends the motion capability including lying down on the floor and standing up. It can carry load up to 2[kg] at each hands. In the following, we took an example of a task carrying a bar in an environment populated by obstacles. The length, diameter and weight of the bar is 1.8[m], 2.4[cm] and 0.5[kg] respectively. Figure 2 illustrates a real experiment.

3 Task-Driven Support Polygon Reshaping for Reaching

There are many works in the literature that have focused on the generation of whole-body motions for complex mechanisms such as humanoid robots or digital actors. A popular approach for motion specification has been, instead of setting explicitly the value of all degrees of freedom, to only specify the values of a task to be accomplished by a given end-effector. The idea is to benefit from the redundancy of the mechanism to choose the solution that best solves the task according to some constraints. Among these works, inverse kinematics algorithms that project tasks with lower priority into the null space of the Jacobian of the higher priority tasks have been widely studied (e.g., [30,32,33,34,37,36,38]).

Our contribution is to consider the possibility of reshaping the support polygon by stepping to increase the accessible space of the end-effectors in the 3D space. The problem we address can be viewed as a 3D extension of the 2D problem addressed in [31]. In [31] the authors propose a strategy for the control of a pattern generator by monitoring the arm manipulability. While their model lies in the sagittal plane, our approach makes use of the whole body in the 3 directions of the 3D space. Moreover, in spite of our reasoning being based on inverse kinematics and simple geometric support polygon reshaping, our method guarantees that the motion is dynamically stable. This property is a consequence of the pattern generator [11] we use to generate the stepping behavior.

3.1 Method Overview

The support polygon reshaping integrates two important components, the generalized inverse kinematics and dynamic walking pattern generator. Figure 3 shows an overview of the method. The task is specified in the workspace as a desired velocity $\dot{\boldsymbol{x}}_j$ with priority j from which the generalized IK solver computes the whole-body motion as joint velocities $\dot{\boldsymbol{q}}$ of the robot. Meanwhile, several criteria such as manipulability or non singularity are monitored.

As long as the criteria are satisfied, the computation of whole-body motion continues until the target of the task is achieved. If the task cannot be achieved due to unsatisfied criteria, the support polygon planner is triggered in order to extend reachable space. A geometric module determines the direction and position of the deformation of support polygon so that the incomplete task is fulfilled. The position of a foot is then derived to generate the motion of CoM $\dot{\boldsymbol{x}}_{CoM}$ by using a dynamic walking pattern generator [11]. Using this CoM motion, the original task is then redefined as the whole-body motion including stepping that is recalculated using the same generalized IK solver.

Let us first overview the generalized inverse kinematics framework. Then we will show how the support polygon is reshaped.

3.2 Whole-Body Motion Generation Using Generalized Inverse Kinematics

Inverse Kinematics for Prioritized Tasks. Let us consider a task $\dot{\boldsymbol{x}}_j$ with priority j in the workspace and the relationship between the joint angle velocity $\dot{\boldsymbol{q}}$ is described using Jacobian matrix, like $\dot{\boldsymbol{x}}_j = \boldsymbol{J}_j \dot{\boldsymbol{q}}$. For the tasks with the first priority, using pseudoinverse $\boldsymbol{J}_1^{\#}$, the joint angles that achieves the task is given:

$$\dot{\boldsymbol{q}}_1 = \boldsymbol{J}_1^{\#} \dot{\boldsymbol{x}}_1 + (\boldsymbol{I}_n - \boldsymbol{J}_1^{\#} \boldsymbol{J}_1) \boldsymbol{y}_1 \tag{1}$$

where $\boldsymbol{y}_1$, n and $\boldsymbol{I}_n$ are an arbitrary vector, the number of the joints and identity matrix of dimension n respectively.

For the task with second priority $\dot{\boldsymbol{x}}_2$, the joint velocities $\dot{\boldsymbol{q}}_2$ is calculated as follows [30]:

$$\begin{aligned} \dot{\boldsymbol{q}}_2 &= \dot{\boldsymbol{q}}_1 + \hat{\boldsymbol{J}}_2^{\#} (\dot{\boldsymbol{x}}_2 - \boldsymbol{J}_2 \dot{\boldsymbol{q}}_1) + (\boldsymbol{I}_n - \boldsymbol{J}_1^{\#} \boldsymbol{J}_1)(\boldsymbol{I}_n - \hat{\boldsymbol{J}}_2^{\#} \hat{\boldsymbol{J}}_2) \boldsymbol{y}_2 \\ &\text{where} \quad \hat{\boldsymbol{J}}_2 \equiv \boldsymbol{J}_2 (\boldsymbol{I}_n - \boldsymbol{J}_1^{\#} \boldsymbol{J}_1) \end{aligned} \tag{2}$$

where $\boldsymbol{y}_2$ is an arbitrary vector of dimension n. It can be extended to the task of j^{th} $(j \geq 2)$ priority in the following formula [32,33].

$$\begin{aligned} \dot{\boldsymbol{q}}_j &= \dot{\boldsymbol{q}}_{j-1} + \hat{\boldsymbol{J}}_j^{\#} (\dot{\boldsymbol{x}}_j - \boldsymbol{J}_j \dot{\boldsymbol{q}}_{j-1}) + \boldsymbol{N}_j \boldsymbol{y}_j \\ \boldsymbol{N}_j &= \boldsymbol{N}_{j-1} (\boldsymbol{I}_n - \hat{\boldsymbol{J}}_j^{\#} \hat{\boldsymbol{J}}_j), \ \hat{\boldsymbol{J}}_j \equiv \boldsymbol{J}_j (\boldsymbol{I}_n - \hat{\boldsymbol{J}}_{j-1}^{\#} \hat{\boldsymbol{J}}_{j-1}) \end{aligned} \tag{3}$$

The CoM Jacobian [12] can be also included as a task.

Weighted Pseudoinverse. In most cases, it is considered to be preferable for a humanoid robot use the lighter links to achieve tasks. For this purpose, we introduce a weighted pseudoinverse:

$$\boldsymbol{J}_W^{\#} = (\boldsymbol{J}^T \boldsymbol{W} \boldsymbol{J})^{-1} \boldsymbol{J}^T \boldsymbol{W}, \ \boldsymbol{W} = \text{diag}\{\sqrt{W_1}, \dots \sqrt{W_n}\} \tag{4}$$

The weight W_i of each joint is given as the ratio of the mass m_i of the link i to the total mass M, namely m_i/M. Moreover, a selection matrix $\boldsymbol{S} = \text{diag}\{S_1, \dots S_n\}$ ($S_i = 0$ or 1) is multiplied to this inverse to select the activated

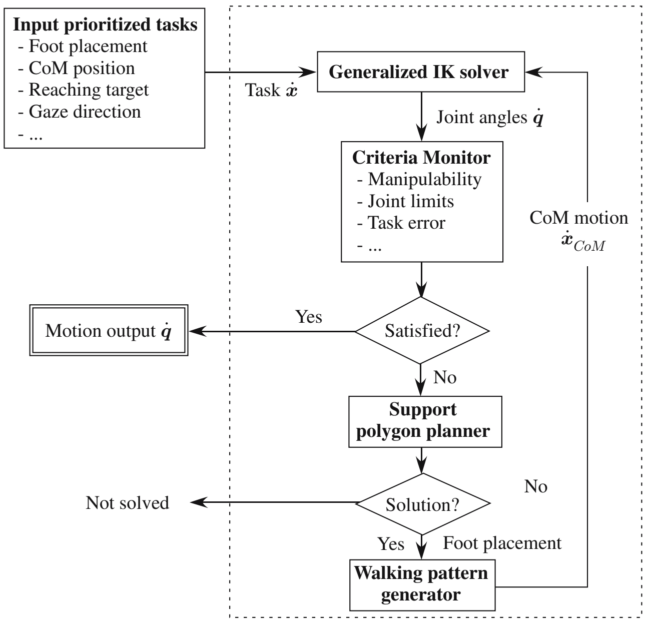


Fig. 3. Method overview

joints according to the task specification. The selection matrix is set to $\boldsymbol{I}_n$ if the all the joints are used to achieve the task.

Using this weighted Jacobian first lighter links are used then heavier ones. By combining a selection matrix $\boldsymbol{S}_l$ that forbids using the joints approaching the limit of the movable range, the heuristics of whole-body motion workspace extension [38] can be implemented in a simpler way.

Monitoring Task Execution Criteria. While the motion is being computed by the generalized IK, several properties are monitored.

One of the important measures is the manipulability [35] defined as:

$$w \equiv \sqrt{\det\{\boldsymbol{J}\boldsymbol{J}^T\}} \tag{5}$$

This measure is continuously tracked during the motion generation as well as others such as joint angle limits or end-effector errors from the target. If it becomes below a certain value, it means that it is difficult to achieve the task.

Joint limit constraint can be taken into account by introducing another selection diagonal matrix $\boldsymbol{S}_l$ whose i^{th} component become zero if the corresponding joint reaches a limit angle.

As shown in Fig. 3, when the monitor detects the measures that prevent the task from being achieved, the support polygon reshaping is launched to extend the accessible space as detailed in the next section.

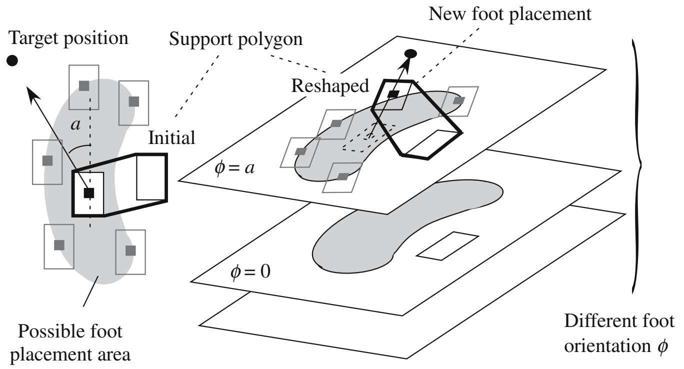


Fig. 4. Support polygon reshaping method

3.3 Support Polygon Reshaping

Figure 4 shows the proposed support polygon reshaping scheme. This simple algorithm allows the humanoid robot to make a step motion, keeping a large margin of accessible area for the task by facing the upper body to the target direction.

Then the CoM motion $\dot{\boldsymbol{x}}_{CoM}$ is computed from the new foot position by the walking pattern generator based on preview control of ZMP [11]. The basic idea is to calculate the CoM position and then the leg motions by anticipating the desired future ZMP positions that correspond to the footsteps.

Finally the original task is redefined as another problem of whole-body task using this newly generated CoM motion. The same generalized IK solver framework introduced in 3.2 is used to incorporate the motion required for task and the stepping motion in the whole-body level.

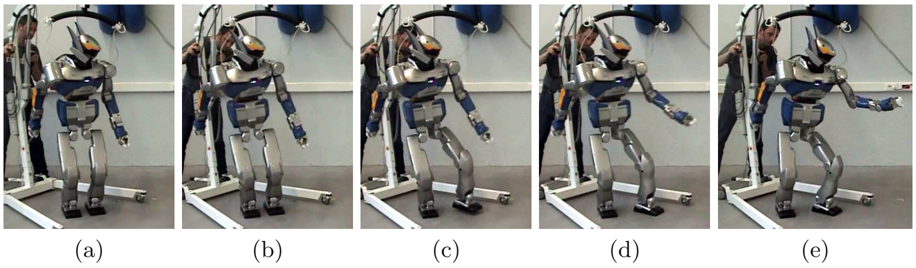


Fig. 5. Experimentation on HRP-2. Putting weight on the right foot in (b), the robot goes through a posture that is not statically stable (c) to finish stepping in (c). The final goal of the end effector is achieved at (e). Notice that the robot makes a whole-body motion including reaching task, stepping and keeping the gaze.

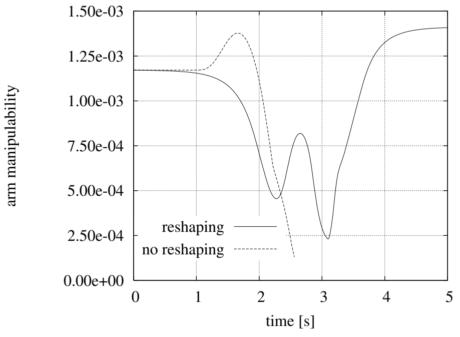


Fig. 6. Without support polygon reshaping, the manipulability measure decreases below the threshold. Although it also decreases with reshaping, the manipulability increases in the course of stepping motion.

3.4 Experimental Results

In the following experiment, the humanoid robot is required to reach a position with the left hand. Four tasks are given with the following priority (i) foot placement, (ii) CoM position, (iii) hand reaching task and (iv) gaze direction in the order of higher priority. For all the tasks the weighted Jacobian (4) is utilized for inverse kinematics. As for the selection matrix $\boldsymbol{S}$, all the degrees of freedom are used, namely setting $\boldsymbol{S}$ to $\boldsymbol{I}_n$, for all the tasks. The reaching task is defined by the target positions without specifying orientation of the hand.

The monitored criteria here during the motion are the manipulability of the arm and the error between the reference end-effector position and the one calculated by the IK solver. The robot tries to reach the target first with the CoM position at the center of the initial support polygon. If those values go below a certain threshold, the support polygon reshaping process is activated. Here the thresholds of manipulability and end-effector error are empirically set to 1.5×10^{-4} and 4.0×10^{-5} [m] respectively.

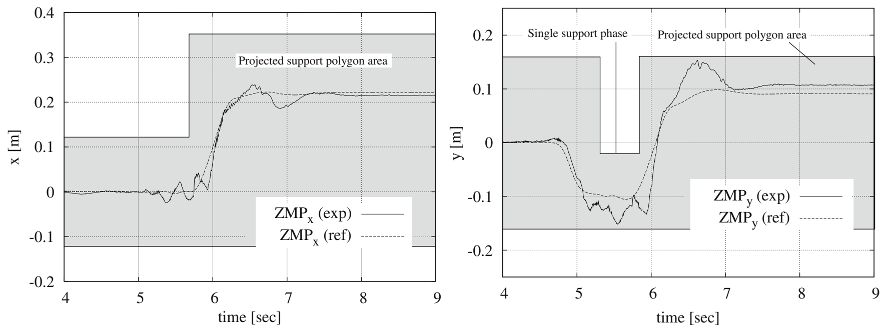


Fig. 7. Evolution of the ZMP coordinates during the motion. The shaded area expresses the transition of the projection of support polygon in x axis (left) et in y axis (right).

Figure 6 shows the manipulability measure of the arm during the reaching task illustrated in Fig. 5. Without reshaping, the arm would approach a singular configuration where the manipulability becomes lower than the threshold at 2.6[s]. The computation keeping the same support polygon is discarded. The reshaping starts at this moment to recalculate the overall whole-body motion including stepping. We can see the manipulability regains higher value at the final position.

In Figure 7, the time development of x and y positions of ZMP measured from the ankle force sensors are plotted for the sideways reaching motion. The dotted and solid lines are the planned and measured trajectories respectively. The shaded areas in those graphs depict the transition of support polygon area projected on x and y axis. As we can see, the planned trajectories of ZMP always stay inside the support polygon.

4 Conclusion

The goal of this paper was to present the work in progress performed by the authors in humanoid robotics. If Robotics and Computer Animation follow different goals with respect to their respective application fields, we think that the research in both areas should benefit from a synergetic view. The synergy comes from a common objective aiming at better understanding the computational issues of human motions. The questions addressed in this paper (how to combine manipulation and locomotion tasks? how to enlarge the scope of redundant system based methods?) are generic questions challenging for both servicing robotics and game industry.

References

1. Yoshida, E., Esteves, C., Sakaguchi, T., Laumond, J.-P., Yokoi, K.: Smooth Collision Avoidance: Practical Issues in Dynamic Humanoid Motion. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 208–213 (2006)
2. Yoshida, E., Kanoun, O., Esteves, C., Laumond, J.-P., Yokoi, K.: Task-driven Support Polygon Reshaping for Humanoids. In: Proc. IEEE-RAS Int. Conf. on Humanoid Robots (Humaniods 2006), pp. 827–832 (2006)
3. Harada, H., Kajita, S., Saito, H., Morisawa, M., Kanehiro, F., Fujiwara, K., Kaneko, K., Hirukawa, H.: A Humanoid robot carrying a heavy object. In: Proc. 2005 IEEE Int. Conf. on Robotics and Automation (2005)
4. Nishiwaki, K., Fukumoto, Y., Kagami, S., Inaba, M., Inoue, H.: Object Manipulation By Hand Using Whole-body Motion Coordination. In: Proc. 2005 IEEE Int. Conf. on Mechatronics and Automation (2005)
5. Yoshida, E., Hugel, V., Blazevic, P.: Pivoting Manipulation of a Large Object: A Study of Application using Humanoid Platform. In: Proc. 2005 IEEE Int. Conf. on Robotics and Automation, pp. 1052–1057 (2005)
6. Michel, P., Chestnutt, J., Kuffner, J., Kanade, T.: Vision-Guided Humanoid Footstep Planning for Dynamic Environments. In: Proc. 2005 IEEE-RAS International Conference on Humanoid Robots (2005)

7. Gutmann, S., Fukuchi, M., Fujita, M.: A Modular Architecture for Humanoid Robot Navigation Steffen Gutmann. In: Proc. 2005 IEEE-RAS International Conference on Humanoid Robots (2005)
8. Okada, K., Ogura, T., Haneda, A., Fujimoto, J., Gravot, F., Inaba, M.: Humanoid Motion Generation System On HRP2-JSK For Daily Life Environment. In: Proc. 2005 IEEE Int. Conf. on Mechatronics and Automation (2005)
9. Yokoi, K., Sian, N.E., Sakaguchi, T., Arisumi, H., Yoshida, E., Kawai, Y., Maruyama, K., Yoshimi, T., Stasse, O., Kajita, S.: Humanoid Robot HRP-2 No.10 with Human Supervision. In: Proc. 36th Int. Conf. on Robotics (2005)
10. Vukobratović, M., Borovac, B.: Zero-Moment Point - Thirty-five Years of its Life. Int. J. Humanoid Robotics 1(1), 157–174 (2004)
11. Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., Hirukawa, H.: Biped Walking Pattern Generation by using Preview Control of Zero-Moment Point. In: Proc. 2003 IEEE International Conference on Robotics and Automation, pp. 1620–1626 (2003)
12. Sugihara, T., Nakamura, Y., Inoue, H.: Realtime Humanoid Motion Generation through ZMP Manipulation based on Inverted Pendulum Control. In: Proc. 2002 IEEE International Conference on Robotics and Automation, pp. 1404–1409 (2002)
13. Sentis, L., Khatib, O.: Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. Int. J. of Humanoid Robotics 2(4), 505–518 (2005)
14. Yoshida, E., Belousov, I., Esteves, C., Laumond, J.-P.: Humanoid Motion Planning for Dynamic Tasks. In: Proc. 2005 IEEE-RAS International Conference on Humanoid Robots, pp. 1–6 (2005)
15. Gleicher, M.: Comparing Constraint-Based Motion Editing Method. Graphical Models 63, 107–134 (2001)
16. Kuffner, J., Kagami, S., Nishiwaki, K., Inaba, M., Inoue, H.: Dynamically-stable motion planning for humanoid robots. Autonomous Robots 1(12), 105–118 (2002)
17. Kuffner, J., Nishiwaki, K., Kagami, S., Inaba, M., Inoue, H.: Motion planning for humanoid robots. In: Proc. 20th Int. Symp. Robotics Research (2003)
18. Chestnutt, J., Kuffner, J.: A Tiered Planning Strategy for Biped Navigation. In: Proc. IEEE Int. Conf. on Humanoid Robotics (2004)
19. Stilman, M., Kuffner, J.: Navigation among movable obstacles: Real-time reasoning in complex environments. In: Proc. IEEE Int. Conf. on Humanoid Robotics (2004)
20. Shiller, Z., Yamane, K., Nakamura, Y.: Planning Motion Patterns of Human Figures Using a Multi-layered Grid and the Dynamics Filter. In: Proc. of 2001 IEEE International Conference on Robotics and Automation, pp. 1–8 (2001)
21. Kanehiro, F., Hirukawa, H., Kaneko, K., Kajita, S., Fujiwara, K., Harada, K., Yokoi, K.: Locomotion Planning of Humanoid Robots to Pass through Narrow Spaces. In: Proc. 2004 IEEE Int. Conf. on Robotics and Automation (2004)
22. Okada, K., Ogura, T., Haneda, A., Kousaka, D., Nakai, H., Inaba, M., Inoue, H.: Integrated System Software for HRP Humanoid. In: Proc. 2004 IEEE Int. Conf. on Robotics and Automation, pp. 3207–3212 (2004)
23. Okada, K., Haneda, A., Nakai, H., Inaba, M., Inoue, H.: Environment Manipulation Planner for Humanoid Robots Using Task Graph That Generates Action Sequence. In: Proc. 2004 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 1174–1179 (2004)
24. Yoshida, E.: Humanoid Motion Planning using Multi-Level DOF Exploitation based on Randomized Method. In: Proc. 2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 25–30 (2005)
25. Yamane, K., Kuffner, J., Hodgins, J.: Synthesizing animations of human manipulation tasks. In: ACM Trans. on Graphics (Proc. SIGGRAPH 2004) (2004)

26. Esteves, C., Arechavaleta, G., Pettré, J., Laumond, J.-P.: Animation Planning for Virtual Characters Cooperation. ACM Trans. on Graphics 25(2), 319–339 (2006)
27. Parent, R.: Computer Animation: Algorithms and Techniques. Morgan Kaufmann Publishers, San Francisco (2002)
28. Kanehiro, F., et al.: Virtual humanoid robot platform to develop controllers of real humanoid robots without porting. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 1093–1099 (2001)
29. Kaneko, K., et al.: The Humanoid Robot HRP-2. In: Proc. IEEE/RSJ Int. Conf. on Robotics and Automation, pp. 1083–1090 (2004)
30. Nakamura, Y.: Advanced Robotics: Redundancy and Optimization. Addison-Wesley Longman Publishing, Boston (1991)
31. Yoshida, H., et al.: Mobile manipulation of humanoid robots - a method of adjusting leg motion for improvement of arm's manipulability. In: Proc. IEEE/ASME Int. Conf. on Advanced Intelligent Mechatronics, pp. 266–271 (2001)
32. Siciliano, B., Slotine, J.-J.E.: A general framework for managing multiple tasks in highly redundant robotic systems. In: Proc. IEEE Int. Conf. on Advanced Robotics, pp. 1211–1216 (1991)
33. Baerlocher, P., Boulic, R.: An inverse kinematics architecture enforcing and arbitrary number of strict priority levels. The Visual Computer 20, 402–417 (2004)
34. Yamane, K., Nakamura, Y.: Natural motion animation through constraining and deconstraining at will. IEEE Trans. on Visualization and Computer Graphics 9(3), 352–360 (2003)
35. Yoshikawa, T.: Manipulability of Robotic Mechanisms. Int. J. Robotics Research 4(2), 3–9 (1985)
36. Liegois, A.: Automatic supervisory control of the configuration and behavior of multibody mechanisms. IEEE Trans. on Systems, Man, and Cybernetics 7, 868–871 (1977)
37. Sentis, L., Khatib, O.: A whole-body control framework for humanoids operating in human environments. In: Proc. IEEE Int. Conf. on Robotics and Automation, pp. 2641–2648 (2006)
38. Sian, N.E., et al.: A Framework for Remote Execution of Whole Body Motions for Humanoid Robots. In: Proc. IEEE/RAS Int. Conf. on Humanoid Robots, pp. 608–626 (2004)

Conveying Emotions through Facially Animated Avatars in Networked Virtual Environments

Fabian Di Fiore, Peter Quax, Cedric Vanaken,
Wim Lamotte, and Frank Van Reeth

Hasselt University - tUL - IBBT
Expertise Centre for Digital Media
Wetenschapspark 2
BE-3590 Diepenbeek
Belgium
{fabian.difiore,peter.quax,cedric.vanaken,
wim.lamotte,frank.vanreeth}@uhasselt.be
http://www.edm.uhasselt.be

Abstract. In this paper, our objective is to facilitate the way in which emotion is conveyed through avatars in virtual environments. The established way of achieving this includes the end-user having to manually select his/her emotional state through a text base interface (using emoticons and/or keywords) and applying these pre-defined emotional states on avatars. In contrast to this rather trivial solution, we envisage a system that enables automatic extraction of emotion-related metadata from a video stream, most often originating from a webcam. Contrary to the seemingly trivial solution of sending entire video streams — which is an optimal solution but often prohibitive in terms of bandwidth usage — this metadata extraction process enables the system to be deployed in large-scale environments, as the bandwidth required for the communication channel is severely limited.

Keywords: Facial animation, emotions, avatars, MPEG-4, networked virtual environments, immersive communication.

1 Introduction

Motivation. Conveying an emotional state between users in a virtual environment is an essential component in achieving total immersion in a three-dimensional world. The currently existing systems provide unsatisfactory results, as they require quite a bit of user intervention in order to keep the avatar's state in sync with the actual emotional state of the user. While this manual process suffices for a coarse impression of the emotional state of the correspondent, reality is that emotional state changes rapidly and in a non-discrete fashion. To keep up with the pace in which the parameters change over the course of a conversation, an automatic metadata extraction and visualisation system for emotional data is clearly required.

© Springer-Verlag Berlin Heidelberg 2008

Contribution. The main contribution of this work is to extract the emotion-related metadata from real-time video streams — most often captured through a webcam pointed towards the user's face — and applying these onto a stylised representation of an avatar. To realise the goals stated in the motivation, we developed a hybrid approach combining benefits of facial animation and user-controlled 2D modelling and animation techniques. Facial animation is employed to extract the movement and timing of the user's facial components, lessening the need for high-bandwidth channels typically required by video-enabled applications. In the visualisation phase, we opt for a structured 2D methodology as the face to which the captured facial movements are applied can be either drawn by hand or based on real footage. Another contribution to the research already carried out is the integration of these types of information in (large-scale) 3D environments.

Approach. Technically, the challenge is to extract precisely the right part of information from a sequence of input frames required for conveying the emotional information. Whilst just transmitting the entire sequence of video frames would be an intuitive way of solving the problem, this is often prohibitive in terms of bandwidth consumption, especially when dealing with 3D environments containing large numbers of users. Therefore, the emotional data is extracted using face and feature extraction algorithms, and only the required parameters (in the form of a set of MPEG-4 feature coordinates) are transmitted over the network. At the receiving side, the parameters are used to animate the avatar representing the originating side.

Paper Organisation. This paper is organised as follows. We start with an overview of related work in the field including realistic approaches, techniques adhering to 2D, approaches which explicitly exploit 3D geometries and some pointers to related work in the field of networked virtual environments (Section 2). Subsequently, the different components making up our system are described in detail in Section 3. We conclude this paper with some clarifying results and our conclusions.

2 Related Work

In this section we look at some existing work that is related to the topic in question, both from the viewpoints of stylised animation and immersive communication in networked virtual environments.

2.1 Facial Animation

Towards Realism. Starting with Parke [1], many researchers have explored the field of realistic facial modelling and animation. For the modelling part this has led to the development of diverse techniques including physics based muscle modelling, the use of free-form deformations, and the use of spline muscle models. For the animation part, the complexity of creating life-like character animations led to performance-driven approaches such as motion capturing and motion retargeting.

We limit this discussion to published work employing some of the mentioned techniques targeted at modelling and/or animating 2D faces. Fidaleo et al. presented a facial animation framework based on a set of Co-articulation Regions (CR) for the control of 2D animated characters [2]. CRs are parameterised by muscle actuations and are abstracted to high-level descriptions of facial expression. Bregler et al. use capturing and retargeting techniques to track motion from traditionally animated cartoons and retarget it onto new 2D drawings [3]. That way, by using animation as the source, similar-looking new animations can be generated.

Although the described techniques are promising and deliver very appealing results, major issues can be identified when targeted to avatars. In the animation stage, they don't offer much freedom of exaggeration (e.g., extensive 3D modelling) whereas the modelling stage implicates a lot of tedious and cumbersome work for the animator (e.g., placing physical markers).

Sticking to 2D. In 1996, Kristinn Thórisson described a dedicated facial animation system, *'ToonFace'*, that uses a simple scheme (a face gets divided into seven main features) for generating facial animation [4]. Ruttkay and Noot discuss *'CharToon'* which is an interactive system to design and animate 2D cartoon faces [5]. Despite its wide range of potential applications (faces on the web, games for kids, ...) a major drawback compared to our approach is that transformations outside the drawing plane are not supported.

Towards 3D. Recently popular, non-photorealistic rendering (NPR) [6,7] techniques (in particular, 'Toon Rendering') are used to automatically generate stylised cartoon renderings. Starting from 3D geometrical models, NPR techniques can generate stylised cartoon renderings depicting outlines with the correct distortions and occlusions. In order to introduce more concepts of 2D animation, Paul Rademacher presented a view-dependent model wherein a 3D model changes shape based on the direction it is viewed from [8].

Starting from 3D has the advantage of the possibility to automatically create extreme frames but at the cost of heavy modelling and, in addition, the results suffer from being too '3D-ish' when one pursues the typical liveliness of 2D animation.

2.2 Immersive Communication

Communication between users in a virtual environment was traditionally done through the use of text-based chat, or, in most recent examples, through real-time VOIP sessions. Obviously, the application of video communication in this context would be an ideal solution, as the immersion in the 3D world remains at an optimal level — not requiring the user to be distracted by the use of cumbersome input devices such as a keyboard.

Various ways to integrate video into an NVE-like application can be envisioned. For example, in [9], a video recording is made of a person turning 360 degrees in front of a camera setup. Each time the avatar needs to be displayed in virtual reality, a set of two images is selected from the obtained sequence. The use of two distinct images is needed for stereo visualisation in a CAVE environment. Of course, the images being displayed are static, and only their position in the scene is subject to change.

The authors of [10] also use an immersive display technology, and use the positional data gathered from an electromagnetic tracking device to place the avatar in virtual space. Stereo-image video cameras are placed in the corners of the immersive display setup to capture moving images of the user together with depth information, in sync with the captured motion information. All data is subsequently transmitted to the other clients, which choose those frames from the sequence that were captured with the stereo camera that matches best with the position of the avatar from their viewpoint. While the system yields good results, it is only applicable in small-scale deployment due to high deployment costs and bandwidth issues, as data from every camera has to be transmitted to each client.

In [11], (full body) video captured from a camera is applied to a three-dimensional mesh of a human figure. Some enhancements are made to provide additional features beside pure video (comments and gestures). The system described is used mainly for guiding users through vast virtual spaces. We should point out here that this system can only be used off-line (i.e. with pre-recorded sequences).

Finally, [12] uses an off-line reconstructed head model to project real-time video onto. Again, the setup used is an immersive display setup and obtains positional data from tracking devices. Segmentation of the video images is facilitated as only information on the user's head is relevant. As the system was designed for use in a controlled local area network environment, no encoding and/or decoding of video sequences was needed, optimising the quality obtained. Other examples of related work are presented in [13,14,15].

Simplified implementations of similar technology often come packaged with webcams, such as those made by Logitech. It is important to note that these types of software only work with specific hardware and often use simplified methods to extract facial information (e.g., histogram- or colour-based methods). Also, there is no provisioning for transmission of the relevant feature data in any appropriate standard.

3 Our Approach

The established ways of conveying emotions between users in virtual environments vary from just transmitting the entire sequence of video frames, over performance-driven animation, to manually select one's emotional state. This is often prohibitive either in terms of bandwidth consumption or in terms of manual input.

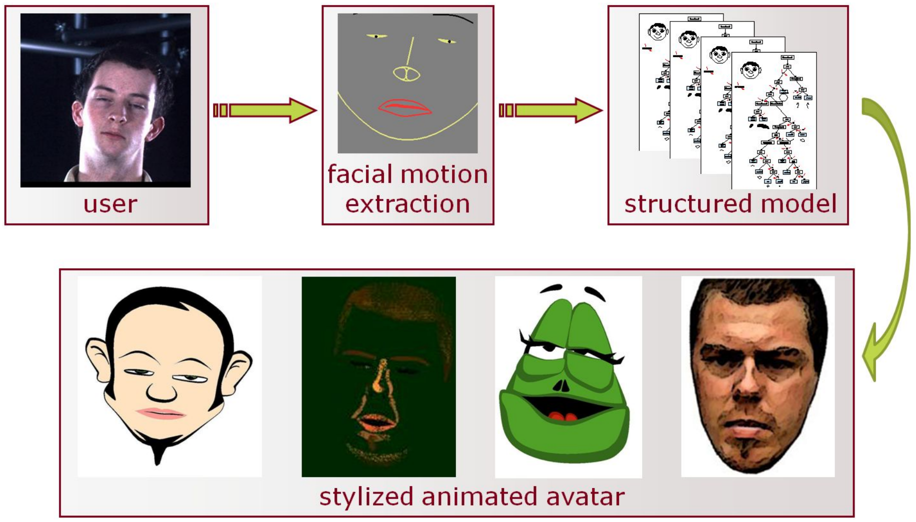


Fig. 1. Overview of the different components of creating and animating an avatar

In our approach (Figure 1) the emotional data is extracted automatically using face and feature extraction algorithms, and only the required parameters (in the form of a set of MPEG-4 feature coordinates) are transmitted over the network. At the receiving side, the parameters are used to animate the avatar representing the originating side. The novelty of our approach lies in how we combine benefits from performance-driven facial animation and user-controlled 2D modelling and animation techniques. Performance-driven facial animation is employed to extract the movement and timing of the user's facial components, lessening the need for high-bandwidth channels typically required by video-enabled applications. In the visualisation phase, we opt for a structured 2D methodology as the face to which the captured facial movements are applied can be either hand drawn or incorporated real footage. Another contribution to the research already carried out is the integration of these types of information in (large-scale) 3D environments.

3.1 Avatar Creation

Modelling Extreme Poses of Drawn Facial Parts. Instead of drawing a 'complete' face at once, every individual part (face outlines, mouth, nose, left eye, right eyebrow, . . .) of the face can be drawn independent of the others. These facial components (also denoted facial channels) are arranged in a hierarchical manner, defined as Hierarchical Display Model (HDM).

In order to achieve convincing 3D-like animations, several view-dependent versions of the HDM (each depicting the same face but as seen from a different viewpoint) can be drawn in order to cover out-of-the-plane animation [16]. Considering facial animation from an artistic point of view, realistic behaviour

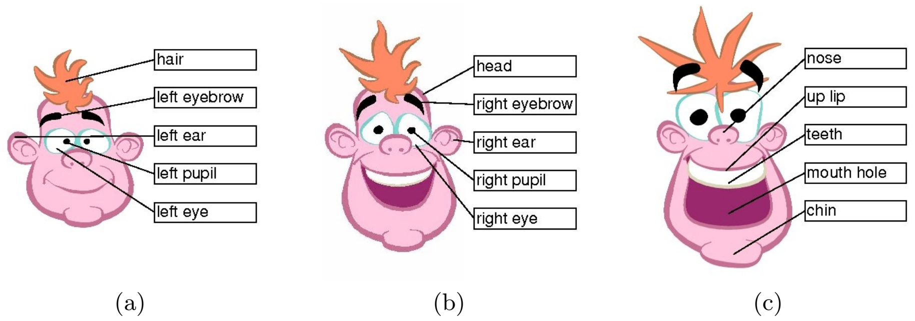


Fig. 2. Some extreme poses of a drawn animation character composed of only 15 facial channels depicting three expressive versions: (a) $e_{neutral}$, (b) $e_{emotional}$, and (c) $e_{exaggerated}$

is not always desired but there's a need for fake, yet very impressive or dramatic effects; especially when applied to avatars [17,18,19]. Hence, in addition several 'expressive' versions of each facial channel can be modelled covering the range of expressiveness held in the user's mind. So, for each expression type, all channels have a separate version. Figure 2 shows three extreme poses of a drawn animation character illustrating the discussed concepts: (a) is composed of 15 facial channels which all depict the same expressive version $e_{neutral}$ whereas (b) and (c) are made up of the same facial channels illustrating expressive versions $e_{emotional}$ and $e_{exaggerated}$. Typically, in total 18 to 27 extreme poses are more than sufficient to cover a wide range of views and expressions (9 depicting the several views, multiplied by 2 or 3 expressive versions).

Modelling Extreme Poses of Real Footage. Besides freely drawing extreme poses/frames starting from a blank canvas, our system also includes the possibility to create extreme frames by incorporating scanned drawings or real images depicting extreme poses (see Figure 3). Starting from incorporating a real image, depicting one extreme pose, the user can define mesh structures over certain image parts that contain interesting information. In fact these meshes can be grouped/layered together in the usual way such as the Hierarchical Display Model (HDM). First, one or more initial meshes are created (using subdivision surfaces) for only one image, corresponding to one extreme frame. Then, other extreme frames are created by incorporating new images (each depicting another extreme pose) for which the user only has to modify a copied instance of the initial meshes. As a result multiple HDMs can easily be created, where each HDM again corresponds to a specific view.

3.2 Facial Motion Data Capture and Extraction

Facial motion data is directly captured from the user's movements using off-the-shelf hardware such as low-cost webcams and digital cameras. Unlike the rigid demands posed on the frame grabbing process by real-time video (i.e. a minimum

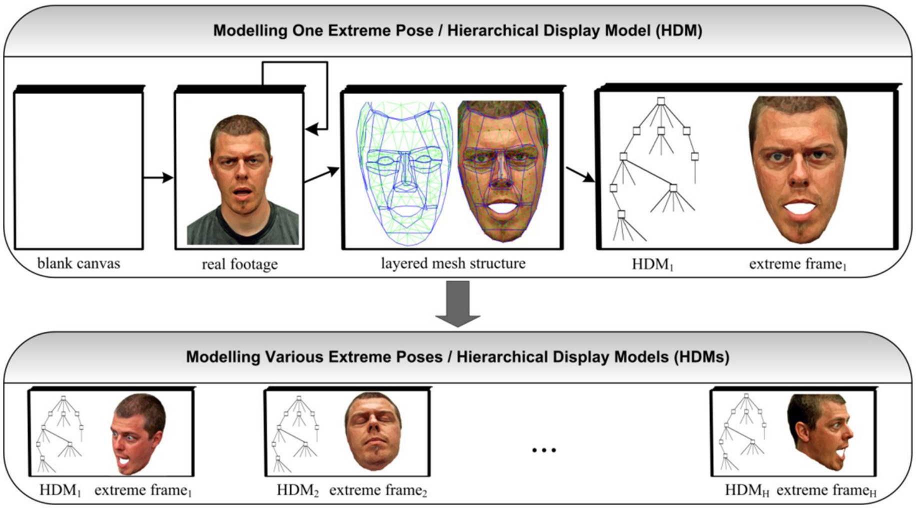

Modelling One Extreme Pose / Hierarchical Display Model (HDM)
blank canvas
real footage
layered mesh structure
HDM$_1$
extreme frame$_1$
Modelling Various Extreme Poses / Hierarchical Display Models (HDMs)
HDM$_1$ extreme frame$_1$
HDM$_2$ extreme frame$_2$
...
HDM$_H$ extreme frame$_H$

Fig. 3. Overview of the different components of creating an avatar by incorporating scanned drawings or real images depicting extreme poses

of 10 fps for fluent motion), which is required for full-frame avatar reconstruction, our solution demands only a few frames to be grabbed in each time frame to achieve adequate results. To substantiate this claim, we remind the reader that the latter technique is much more computationally expensive (i.e. to reconstruct in-between frames in a video sequence) than to interpolate between a very limited set of (feature) coordinates representing the facial expressions on a 3D model.

After the raw video frames are captured, we use the face detection algorithms present in the OpenCV library, which in effect uses techniques based on Haar-like features. Because the existing Haar classifiers are complementary, we combined the results into a set of (possibly) overlapping rectangles, the union of which is calculated in a subsequent step. This step of the process ends with the creation of a set of rectangles representing the detected faces. Elementary image processing algorithms are applied to the detected regions in order to determine the location of the important features — which, for emotion recognition, are mainly the shape of the mouth and the eye/eyebrow combinations. We also exploit some well-known anatomical facts that help to speed up the processing, such as the assumption that there is a minimal distance between the features and their relative position with regards to one another. What we end up with is a black and white mask, for which it is easy to extract the required feature parameters (see Figure 4).

More in detail, data describing the movement of facial components is extracted [20] and made available on a multi-level basis according to the MPEG-4 characterisation [21]: (i, low level) movement of individual feature point positions relative to a set of facial invariant points according to the MPEG-4 Facial Feature Points Location; (ii, medium level) movement of defined areas of the face

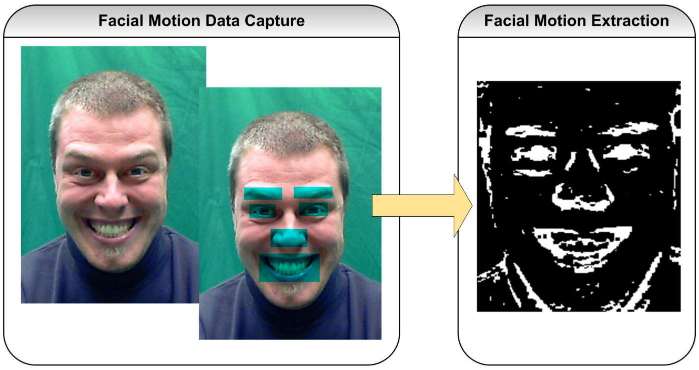


Fig. 4. Overview of the process of facial motion data capture and extraction

described in terms of MPEG-4 Facial Animation Parameters (FAPs); and (iii, high level) motion in terms of MPEG-4 Facial Expressions.

3.3 Animation System

After the extreme poses of the facial parts are modelled, the extracted facial motion data can be applied to animate the avatar. As the extracted motion of the facial components is made available on a multi-level basis, various mappings can be defined between the modelled facial channels and the extracted motion data.

At the lowest level, for each facial channel any arbitrarily control point or user-selected part of the channel can be enforced to inherit the motion of one of the captured MPEG-4 Facial Feature Points. At a medium level, each facial channel can be driven by one or more of the captured MPEG-4 Facial Animation Parameters (FAPs). At the highest level, 'expressive' versions of facial channels can be grouped together on the basis of expressing the same emotion (e.g., joy or sadness). We define these groups as Facial Expression Channels (FECs). These are analogous to the captured MPEG-4 Facial Expressions and can be considered as groupings of FAPS expressing a specific emotion. At the medium level, for instance, user defined facial parts can be driven by one or more of the captured MPEG-4 Facial Animation Parameters (FAPs). This happens in an easy and interactive way during the modelling stage and requires only a reasonable amount of manual input. For each facial part, the animator only has to define regions (FAP regions) using a lasso tool and attribute each of them to a desired FAP. Figure 5 depicts a FAP region defined by the animator. Note that each desired FAP region only has to be defined once for one of the extreme frames. The selection automatically gets propagated to the other extreme frames.

Once the desired mappings have been made, the extracted facial motion data is loaded into the animation system and all keyframes are automatically set, hence, driving the animation. As mentioned in previous section, our solution demands only a few frames to be grabbed in each time frame to achieve adequate results. Missing (i.e. non-captured) frames are reconstructed on-the-fly either by

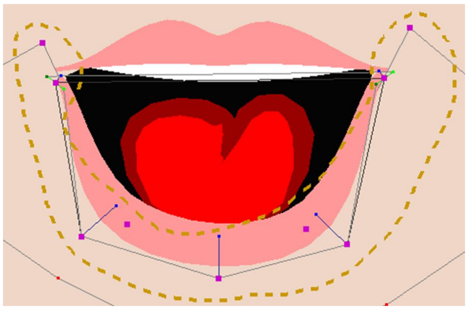


Fig. 5. Example of a user defined FAP region

extrapolating or by interpolating between surrounding frames (after delaying the stream a few frames). For the case of drawn facial parts, the underlying curves making up the drawings are interpolated to create in-between drawings whereas for real footage all corresponding meshes imposed on the keyframes are warped to each other automatically across intermediate frames. We refer the reader to [22] for an in-depth explanation of the animation system.

3.4 Visualisation of Results in 3D Environments

While we won't go into great detail on the network architecture behind the networked virtual environment that is used to test the results, we will mention some of the features that show that the integration of this type of information can be done with relative ease. The networked virtual environment being used is based on a client/server architecture, with an intermediate layer of 'proxy' servers that are employed to channel the data flows and which mitigate the need for multiple (persistent) connections between a client and the servers responsible for state management (here referred to as the 'logic' servers). As the architecture is already designed with the ability to efficiently channel all positional data related to avatars (as is true for all NVE architectures), it becomes a trivial extension to include the emotional data that is generated by our system. In effect, the main difference between the two types of information is in their update rate (which is much higher for positional updates) and their relative importance. While the data required for conveying emotions can be considered non-essential, the same is not true (to an extent) for positional data.

4 Results

Figure 6 depicts some stills of a generic scene consisting of 3 billboards where a user is, at the same time, visualised using the entire video stream (first billboard) requiring large amounts of bandwidth, and two avatar forms including the emotional information (requiring minimal additional bandwidth). The second billboard depicts the emotion conveyed by the first billboard but retargeted

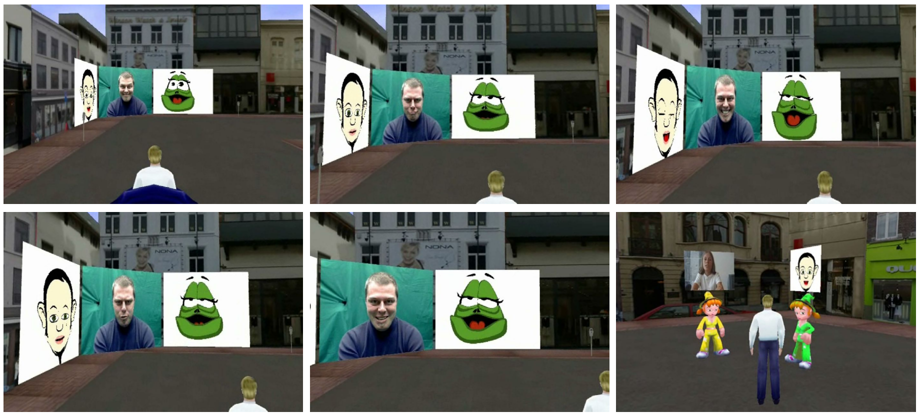


Fig. 6. Snapshots of a generic scene consisting of 3 billboards. The first billboard visualises the input data whereas the second and third depict the emotion conveyed by the first billboard but retargeted to a drawn man's face and a drawn frog's face. The last image shows two users communicating through their avatars.

to an expressive drawn man. For this avatar 18 extreme frames were used consisting of 9 versions which are used to cover different views multiplied by 2 emotional versions (i.e. either neutral or happy) which have been drawn for each view-dependent one. The model itself consists out of 15 facial parts and in total 33 FAP regions were defined. The third billboard shows some snapshots of the same emotions retargeted to a very expressive (i.e. always happy) drawn frog. In this example, only 4 extreme frames were used to drive the animation. The frog's face is composed of 14 facial channels. The last image shows two users communicating through their avatars.

Discussion. All results are driven by real-time gathered facial motion data and have been evaluated visually by the authors. Certainly, the degree of resemblance depends on the quality (i.e. lifelike modelling versus exaggerated modelling) and the size (i.e. number of view-dependent or emotional extreme frames) of the created emotion space. However, in general we can conclude that there is a clear resemblance between the features of the user's emotional expressions and the final animated output.

5 Conclusions

In this paper we have presented a combination of existing technology allowing users to convey emotions through their avatars in a 3D virtual environment, without major impact on the bandwidth requirements. This is achieved by capturing real-time video using off-the-shelf webcam hardware, and analysing the resulting data flows to extract the facial features important for (human) emotion recognition. Instead of using an actual recognition process, we determine the important feature coordinates and send these to the receiving side, where they can be used to animate a (stylised) avatar representation of the user. Technically, this is achieved by representing the coordinates as a set of MPEG-4 facial feature points and combining them with MPEG-4 Facial Animation Parameters. An in-betweening process is used to interpolate the keyframe information that is sent between the communicating parties. The results have been shown to be applicable in a generic 3D environment.

Acknowledgements

Part of the research at EDM is funded by the ERDF (European Regional Development Fund) and the Flemish government. The authors would like to thank Panagiotis Issaris and Jeroen Dierckx for their help.

References

1. Frederic, I.P.: Computer generated animation of faces. In: Proceedings of ACM National Conference, pp. 451–457 (1972)
2. Fidaleo, D., Neumann, U.: CoArt: Co-articulation Region Analysis for Control of 2D Characters. In: Proceedings of Computer Animation (CA 2002), pp. 17–22 (June 2002)
3. Bregler, C., Loeb, L., Chuang, E., Deshpande, H.: Turning to the masters: Motion capturing cartoons. In: Proceedings of SIGGRAPH, July 2002, vol. 21(3), pp. 399–407. ACM, New York (2002)
4. Thórisson, K.R.: Toonface: A system for creating and animating interactive cartoon faces. Technical report, MIT Media Laboratory, Learning and Common Sense 96–01 (April 1996)
5. Ruttkay, Z., Noot, H.: Animated chartoon faces. In: NPAR2000: Symposium on Non-Photorealistic Animation and Rendering, pp. 91–100 (June 2000)
6. Gooch, B., Gooch, A.A.: Non-Photorealistic Rendering. A. K. Peters Ltd. (2001) ISBN: 1568811330
7. Strothotte, T., Schlechtweg, S.: Non-Photorealistic Computer Graphics. Modeling, Rendering, and Animation. Morgan Kaufmann Publishers, San Francisco (2002)
8. Rademacher, P.: View-dependent geometry. In: Rockwood, A. (ed.) Proceedings of SIGGRAPH, Los Angeles, pp. 439–446. ACM, Addison Wesley Longman (1999)
9. Insley, J., Sandin, D., DeFanti, T.: Using video to create avatars in virtual reality. In: Visual Proceedings of the 1997 SIGGRAPH Conference, Los Angeles, CA, p. 128 (1997)
10. Ogi, T., Yamada, T., Tamagawa, K., Hirose, M.: Video avatar communication in a networked virtual environment. In: Proceedings of the 10th Annual Internet Society Conference, vol. Electronic edn. (2000)
11. Yura, S., Usaka, T., Sakamura, K.: Video avatar: Embedded video for collaborative virtual environment. In: Proceedings of the IEEE International Conference on Multimedia Computing and Systems, vol. 2, p. 433 (1999)
12. Rajan, V., Subramanian, S., Keenan, D., Johnson, A., Sandin, D., Defanti, T.: A realistic video avatar system for networked virtual environments. In: Proceedings of IPT 2002, Orlando, FL (2002)

13. Liu, Z., Zhang, Z., Jacobs, C., Cohen, M.: Rapid modeling of animated faces from video. The Journal of Visualization and Computer Animation 12(4), 227–240 (2001)
14. Wang, R.S., Wang, Y.: Facial feature extraction and tracking in video sequences. In: IEEE International Workshop on Multimedia Signal Processing, pp. 223–238 (1997)
15. Quax, P., Flerackers, C., Jehaes, T., Lamotte, W.: Scalable transmission of avatar video streams in virtual environments. In: Proceedings of the 2004 IEEE International Conference on Multimedia and Expo (ICME). IEEE Computer Society Press, Los Alamitos (2004)
16. Fiore, F.D., Schaeken, P., Elens, K., Reeth, F.V.: Automatic in-betweening in computer assisted animation by exploiting 2.5D modelling techniques. In: Proceedings of Computer Animation (CA 2001), pp. 192–200 (November 2001)
17. Blair, P.: Cartoon Animation. Walter Foster Publishing Inc. (1994) ISBN: 1-56010-084-2
18. Barzel, R.: Faking dynamics of ropes and springs. IEEE Computer Graphics and Applications 17, 31–39 (1997)
19. Williams, R.: The Animator's Survival Kit. Faber and Faber Limited, Queen Square London (2001)
20. Vicar, D.M., Ford, S., Borland, E., Rixon, R., Patterson, J., Cockshott, P.: 3D performance capture for facial animation. In: Proceedings of 3D Data Processing, Visualization and Transmission (3DPVT), pp. 42–49 (2004)
21. Pandzic, I.S., Forchheimer, R.: MPEG-4 Facial Animation: The Standard, Implementation and Applications. John Wiley & Sons, Chichester (2002)
22. Fiore, F.D., Reeth, F.V.: Multi-level performance-driven stylised facial animation. In: Proceedings of Computer Animation and Social Agents (CASA 2005), pp. 73–78 (October 2005)

Animating Speech in Games

Scott A. King

Texas A&M University - Corpus Christi
6300 Ocean Drive, Unit 5824
Corpus Christi, Texas 78412, USA
`Scott.King@TAMUCC.edu`
˜http://www.sci.tamucc.edu/ sking

Abstract. Realism in games is constantly improving with increased computing power available to games and as game players demand more visual realism. Therefore, facial animation and particularly animated speech, is becoming more prevalent in games. We present a survey of facial animation techniques suitable for 3D computer games. We break our discussion into two areas: modeling and animation. To model a face, a method of representing the geometry is combined with a parameterization that is used to specify a new shape for that geometry. Changing the shape over time will create animation, and we discuss methods for animating expressions as well as achieving lip-synchronized speech.

Keywords: Facial Modeling, Facial Animation, Animated Speech.

1 Introduction

Facial animation [1, 2, 3] can convey information to game players and increase immersion. Facial expressions [4, 5, 6, 7] can convey emotion, direct attention, or show that user action is required. Animated speech can create a much more immersive experience, but only with high-quality speech synchronization, which is difficult to achieve. This animation required important CPU cycles that would take away from game play and so until recently was rarely used in games. As computers continue to increase in power, and with the advent of powerful programmable GPUs, facial animation in games is becoming more common and expected.

Animated speech requires deformable lips that move in synchrony with the audio. Incorrect visual information can lead to confusion and misinterpretation, so accuracy is important. *Static* realism can be achieved with a high resolution model and texture map. Higher-order surfaces such as NURBS or subdivision surfaces can also increase the static realism at a computational cost. *Dynamic* realism is achieved by creating realistic motion and requires a fast frame rate. Dynamic realism is much more important of the two as it directly affects the effectiveness of conveying the message to the game player through speech.

Facial animation is very important to the entertainment industry leading to development of tools to aid in facial modeling, animation, and speech synchronization. These tools were designed for realistic facial animation and not for

© Springer-Verlag Berlin Heidelberg 2008

speed. Some tools were redesigned and others created to aid the game developer. Some example products are: Face FX$^{\text{TM}}$by OC3 Entertainment (oc3ent.com), Facial Studio for Windows$^{\text{TM}}$from Di-O-Matic (di-o-matic.com), GameFace$^{\text{TM}}$ from aPeerance (dipaola.org/apeerance), Life Mode Interactive's (lifemi.com) LifeStudio:Head$^{\text{TM}}$and LifeStudio:LipSync$^{\text{TM}}$, Magpie Pro of Third Wish Software & Animation (thirdwishsoftware.com), and proFACE from Famous3D (famous3D.com). Also game some engines have built-in facial animation support such as Source$^{\text{TM}}$from valve (source.valvesoftware.com). But, in the entertainment industry products and companies emerge and disappear quickly.

To add facial animation to your game, there are three things to consider: how to represent the geometry, how to parameterize the geometry, and how to animate the geometry; and they are not quite independent. We describe these three areas and discuss solutions that have been proposed in the following sections. We also discuss rendering techniques for increasing realism.

2 Modeling

A facial model combines the face geometry with a parameterization that defines the shape that geometry takes. This could be as simple as a mesh that defines the connectivity of a set of vertices with the location of the vertices making up the parameterization. However, such a large number of parameters is difficult to control. A parameterization that is not directly related to the method of representing the geometry is much more powerful. The parameterization may be used to create different face shapes (i.e. different characters) or to deform the face to change its appearance (i.e. change the expression). Multiple parameterizations may be used together or separately depending on the situation.

2.1 Geometry

For games, the method of representing geometry is limited to those that can be deformed and rendered in real-time. Since graphics hardware work best with triangles, they are an obvious choice, especially low-resolution meshes. With the increase in GPU power and in new functionality, higher-order surfaces, such as subdivision surfaces and NURBS are becoming viable choices.

One tradeoff between quality and speed is the number of vertices. Eyebrows should be present as they are the most important part of the face for showing expression. A single thick line (or possibly three line segments for each brow) that is separate from the rest of the face geometry may be adequate. This will allow for fast dynamic realism. When using triangles, the upper and lower borders of the eyebrows should have at least four or five segments. Using simple material properties with high contrast between the brows and the face such as black, with very little specular light is a good choice. See Figure 1 for example eyebrows. For non-speaking characters, a low-complexity mouth is adequate. For speech, lips need seven segments and should include parts only seen when the mouth is open. Teeth and a tongue [8] are also needed. Figure 2 shows an example of low-resolution lips [9] capable of realistic animated speech.

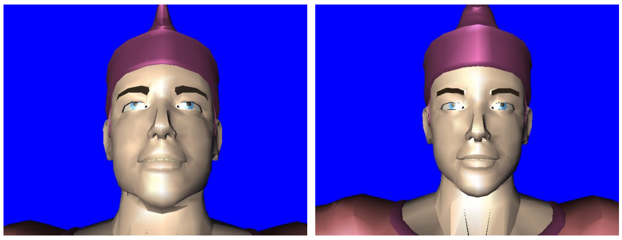


Fig. 1. The image on the left uses 16 triangles (8 will suffice) for each eyebrow, while the image on the right uses 3 lines for each brow

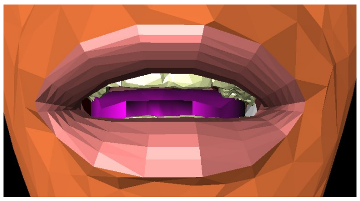


Fig. 2. An example of a low-resolution mouth with lips, teeth and tongue

2.2 Parameterizations

A parameterization abstracts the shape or movement of the surface in $\Re^3$ into something easier to use by an animator or director. Several layers of parameterizations or several separate parameterizations can be used for facial animation. The facial model may be as simple as pure geometry, such as CANDIDE [10], or it could be very complex with separate parameterizations for different parts of the face [11]. The parameterization could be a displacement from a neutral shape [12, 13] or based on muscle movement, sounds, emotions or expressions. Several methods can be used to create animation and each method may define a new parameterization between the animation and the model.

A common early method was to create a parameterization based on empirical study of faces or using techniques from traditional hand-drawn animation [14,15]. A new parameter that interpolates between two extremes poses, or a pose and the neutral face, is often used for each expression. Since muscle contractions actually cause the deformations, using them as a basis for controlling surface deformations has been very popular [16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26]. Muscles are generally given insertion and attachment points with a zone of influence. At the model level, they are a good choice for a parameterization. At the animation level, it is difficult for an animator to specify all the muscle values. Instead, another high-level parameterization, such as emotions or expressions, is often used.

The Facial Action Coding System [27] (FACS) was designed to describe changes in facial appearance. Although the system was designed as a descriptive system, FACS has been embraced as a way to generate facial expressions.

FACS++ [24] was an attempt to fix this shortcoming and an updated FACS (face-and-emotion.com) is now available.

3 Animating

The human face is quite expressive and is used to communicate ideas as well as emotion. Humans interpret faces from birth and are very sensitive to inaccuracies making realistic animation a difficult problem. Animating faces involves speech, expressions (which includes emotion, gestures, etc.), and life motions. Life motions, such as breathing, non-expressive blinking, head motion and hair motion, are all the little things that make the model look alive, but do not convey specific information. Methods for creating 3D facial animation can be broken into three categories: performance capture, keyframing, and physical simulation. These methods are not mutually exclusive and it is possible to combine them.

3.1 Performance Animation

An effective method of creating realistic motion is to capture the motion of an actor giving a performance then apply this motion to a 3D character resulting in extremely realistic motion. Often the motion is captured using video techniques [25,28,29,30]. But other capture methods such as a puppeteer interface [29,31] or magnetic trackers, can be used.

Using captured motion has been a very popular method for achieving animation in games, particularly for animating humans. Facial animation is no exception. Motion capture can give some of the best results for speech synchronization as the effects of prosody and coarticulation are captured.

Performance animation can be used in one of two ways: the final motion for the animation is captured and played back or motion fragments are captured and these are concatenated together to create the final animation. The first method is popular in high-end games due to the good results. However, the results tend to fall into the uncanny valley. It is also very expensive to capture all facial motion in advance. To make the problem tractable, speech is broken into segments (usually phonemes [32,33,34], but triphones or syllables [35] can be used.

Capturing the motion of segments and using those to create novel animations shows promise, but there are some difficulties to overcome. A major problem is a method of time warping [36,37] the motion must be found since the duration of new segments will be different than the duration of captured segments. Another problem is a method to retarget the captured motion data to the model to be animated. Several solutions have been used including using radial basis functions [33], blendshapes [38], and determining muscle activations [39]

There are also some practical issues to deal with such as capturing only non-rigid motion. This is actually a difficult problem as markers are placed on the skin and the skin can move independently of the bone. This means that rigid motion can't be separated from non-rigid motion by using any set of markers.

Using a helmet or fixing the head so as to restrict rigid motion can be used, but these will affect the way the actor speaks. Another problem to overcome is to segment the captured motion correctly. This is a very difficult problem as the timing of phonemes is difficulty to determine. There are automated tools that can be used since the text is known. But their accuracy may not be precise enough for this problem.

When capturing segments, coarticulation and prosody must be dealt with as well. In addition, expressions add complexity. It is also possible to capture texture along with the motion [40,41] to get secondary motion such as wrinkles.

3.2 Keyframing

In order to create animated speech, keyframing is almost a necessity when not using performance capture (e.g. [12,20,21,22,23,31,42,43,44,45,46,47,48,49, 50,51]). This is because the visual signal has to be synchronized with the audio requiring targets at critical times, such as when the lips should be in contact with each other. The differences in using keyframing lie in how the keys are produced, combined, and interpolated.

A common and effective method to create the keyframes is to define an ideal face shape for each possible expression and each phoneme. These keys can be created by changing the model parameters, by having an artist create different key poses, or captured using a 3D digitizer from a sculpture [47] or an actor [32].

Keys can be interpolated in a number of ways: with splines [20,52], giving the keys mass connected with springs [53], linearly, using sine [16], with a puppeteering interface [31,47], as a shape morph [49], and many others. Using a c^2 interpolant is important to get more natural motion. This is particularly true for speech. Methods for interpolating speech keyframes are discussed later.

3.3 Physically-Based Animation

Physics-based models treat the geometry as a point-mass system. Forces are applied to the points and motion is calculated using Newtonian physics. Often, the skin surface is defined as a mesh connected with springs [16,23,26,51,54,55,56,57] [58,59,60,61,62,63,64,65,66]. Applying forces to the nodes via muscle contraction and solving the resulting differential equations gives animation. The integration can be solved using techniques such as fourth-order Runge-Kutta, implicit Euler, or Verlet leapfrog.

3.4 Other Techniques

Many other techniques used for animation can be used effectively in facial animation as well. These techniques make animation more efficient or easier to create (either for an artist, director, or automatically).

Creating a hierarchy of motion, description, surfaces, etc. is a technique that can be used for animating, parameterizing and modeling [11,17,21,22,63,66,67, 68]. Several simple layers replacing a single complex layer can increase performance and make the system easier to use.

Scripting [13, 17, 21, 22, 69, 70, 71, 72] gives greater control over the characters to the director or animator and facilitates the reuse of animation. Procedural animation [19, 73] is quite useful for controlling the many supporting characters in a game and for use in giving the character more natural responses to the game player via artificial intelligence methods. Noise can even be used to create believable facial animation (mrl.nyu.edu/∼perlin/facedemo/).

3.5 Animating Expressions

Expressions convey emotion and concepts and they include head nods, eye blinks, raising eyebrows, eye gaze, and high-level emotions. During communication, expressions [4, 5] are used to accentuate speech, for turn taking, to convey understanding, and to deceive [6]. FACS [27] is quite popular for animating expressions. Expressions can be added explicitly to the animation, automatically based on the emotion of the character [74], based on how the character is participating in the dialogue [75], or from information in the recorded speech [20].

Expressions are formed in three distinct areas [5]: the brows, the eyes, and the mouth. Of these, the most important is the eyebrows for most expressions. It is important that the player can easily see the position of the eyebrows. The eyes, particularly the lower eyelids [6] show the difference between a true emotion and an emotional emblem, or between a true emotion and a false emotion. Expressions such as eyebrow raising and head nods are very important in dialogue management (turn-taking) and greatly enhance the character's believability.

Eye motion [76] is also quite important for realism. Without properly moving eyes, the character will not seem alive. Also, eye contact or lack of eye contact can be used effectively for controlling dialogue with the player. While talking, the head is rarely motionless and this motion is synchronized with speech. Life motions, such as blinking, looking around, and breathing are important secondary motions to add to give your character the illusion of life.

Figure 3 shows a look of surprise at various zooms. Low-resolution models can convey emotion, however, when the character is too far away, expressions are indiscernible and therefore displaying them should be dependent on screen size. Also, without the appearance of wrinkles on the forehead and shadows in the eye sockets it is difficult to determine when the eyebrows are raised.

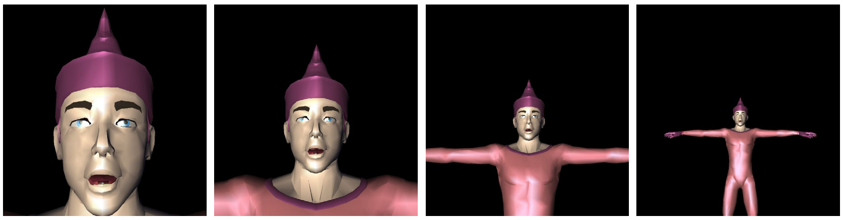


Fig. 3. The same surprise expression from different distances. As fewer and fewer pixels are used to display the face, it becomes much more difficult to recognize.

There is strong evidence [4, 5] that there are universally recognized expressions namely: fear, anger, disgust, sadness, enjoyment, and surprise. Other facial expressions and expressions of emotions may be, and most likely are, cultural specific. For games with an international audience, it is important to not use subtle expressions and to make sure that the expressions are culturally correct.

3.6 Animating Speech

Speech is very difficult to animate correctly. If the velocities of the lips are not realistic, this bad motion will not only detract from the experience, it may also lead to misinterpretation. Traditional animators realized it was too difficult to create by hand and instead just opened and closed the mouth. The audience quickly dispels belief and uses audio only to interpret the utterance, along with visual cues to emotion and facial expression. If the CPU cycles are not available then this simple method of non-synchronization may be used.

The most common approach is to break the speech into phonemes (*phonetic elements* and group similar looking phonemes into visemes[1] placing one or more keyframes per phoneme, and interpolate these keyframes. Linear or spline interpolation, which works well for most animation, falls short when animating speech (mostly because they give incorrect accelerations). Also, a phoneme does not always look the same due to coarticulation and prosody. *Coarticulation* is the effect a phoneme has on, and is affected by, its neighboring phonemes. *Prosodic* features of speech affect the timing and position of the vocal tract parts and include length, accent, stress, tone, and intonation.

If the visual manifestation of coarticulation is missing, important information for the listener will be absent making the speech harder to understand as well as being visually incorrect. Systems using a coarticulation model achieve better results because they allow for different shapes for the same phoneme. There are many different models of coarticulation including look-ahead (rules) [20, 23, 50, 77, 78], cosine interpolation [53], and dominance functions [11, 79, 80, 81, 82, 83].

A few systems have dealt with prosody, mostly in the form of facial expressions, such as eyebrow movements to add emphasis, gaze control for turn-taking, and hand or head motion to reinforce the message. These prosodic effects can come directly from an input speech signal [20, 84] or generated by the system [69, 74]. These expressions are quite helpful in conveying the message to the player.

4 Rendering

Speech can suffer terribly from aliasing as frame rates of 25Hz (or even 100Hz) are not adequate for all situations. The problem arises for consonants where the visible parts of the vocal tract make contact with each other, such as the lips for /p/, /b/, and /m/; the tongue and teeth for /th/ and /d/; or the lips and teeth for /f/ and /v/. When sampling at a regular interval it is quite easy to

[1] The number of different visemes to use varies widely (10-20). Good sources are animator's guides and speech reading literature.

miss the contact entirely or to have it appear too quick. One possible solution is to use irregular intervals, which will happen anyway in game situations as frame intervals are never constant. Aliasing will be reduced but not removed. Instead, these important contact events can be marked and either shifted to occur at the current frame time, or the current time can be set to that time [82].

Modeling and rendering hair [85, 86, 87, 88] and skin [89, 90, 91, 92] are also important, but tend to be quite expensive and beyond the scope of this article. Using texturing to achieve realism is the best choice in most real-time situations. Using programmable GPUs will be key to improving visual realism.

5 Conclusions

Good real-time facial animation techniques are currently available and they will rapidly make their way into games. They are still expensive in terms of computation and in development costs. However, as artificial intelligence techniques continue to improve, the importance of dialogue with the player will necessitate facial animation. As more tools become available and as more research gets converted to commercial use, good facial animation will be commonplace in games.

Acknowledgments. The Harley model is courtesy of nVIDIA from their Cg SDK (developer.nvidia.com).

References

1. Parke, F.I., Waters, K.: Computer Facial Animation. A K Peters (1996)
2. Fleming, B., Dobbs, D.: Animating Facial Features and Expressions. Charles River Media, Inc., Rockland (1999)
3. Osipa, J.: Stop Staring: Facial Modeling and Animation Done Right. SYBEX, Inc., Alameda (2003)
4. Ekman, P.: Darwin and cross cultural studies of facial expression. In: Ekman, P. (ed.) Darwin and Facial Expression: A Century of Research in Review. Academic Press, New York (1973)
5. Ekman, P., Friesen, W.V.: Unmasking the Face. Consulting Psychologists Press, Palo Alto (1984)
6. Ekman, P.: Telling lies: clues to deceit in the marketplace, politics, and marriage. Norton, New York (1985)
7. Faigin, G.: The Artist's Complete Guide to Facial Expression. Watson-Guptill Publications, New York (1990)
8. King, S.A., Parent, R.E.: A 3d parametric tongue model for animated speech. JVCA 12(3), 107–115 (2001)
9. King, S.A., Parent, R.E., Olsafsky, B.: A muscle-based 3d parametric lip model for speech. In: Deformable Avatars, pp. 12–23. Kluwer Academic Publishers, Dordrecht (2001)
10. Rydfalk, M.: Candide: A parameterized face. Technical Report LiTH-ISY-I-0866, Linkoping University, Sweden (October 1987)
11. King, S.A.: A Facial Model and Animation Techniques for Animated Speech. PhD thesis, The Ohio State University, Columbus, OH (June 2001)

12. Bergeron, P., Lachapelle, P.: Controlling facial expressions and body movements in the computer-generated animated short Tony De Peltrie. In: SIGGRAPH 1985 Advanced Computer Animation seminar notes, 1–19 (July 1985)
13. Elson, M.: Displacement facial animation techniques. In: SIGGRAPH 1990 Course Notes, Course 26, State of the Art in Facial Animation, pp. 21–42 (1990)
14. Parke, F.I.: A parametric model for human faces. PhD thesis, University of Utah, Salt Lake City, Utah (December 1974)
15. Nahas, M., Huitric, H., Saintourens, M.: Animation of a B-spline figure. The Visual Computer 3(5), 272–276 (1988)
16. Waters, K.: A muscle model for animating three-dimensional facial expression. In: Computer Graphics (SIGGRAPH 1987 Proceedings), Anaheim, California, vol. 21(4), pp. 17–24 (July 1987)
17. Magnenat-Thalmann, N., Primeau, E., Thalmann, D.: Abstract muscle action procedures for human face animation. Visual Computer 3(5), 290–297 (1988)
18. Guenter, B.: A computer system for simulating human facial expression. PhD thesis, The Ohio State University (1989)
19. Reeves, W.T.: Simple and complex facial animation: Case studies. In: SIGGRAPH 1990 Course Notes 26: State of the Art in Facial Animation, Dallas, Texas, pp. 88–106 (August 1990)
20. Pelachaud, C., Badler, N.I., Steedman, M.: Linguistic issues in facial animation. In: Computer Animation 1991, Tokyo, pp. 15–30. Springer, Heidelberg (1991)
21. Patel, M., Willis, P.J.: FACES — facial animation, construction and editing system. In: Purgathofer, W. (ed.) Eurographics 1991, pp. 33–45. North-Holland, Amsterdam (1991)
22. Patel, M.: Making Faces: The Facial Animation, Construction and Editing System. PhD thesis, School of Mathematical Sciences, University of Bath, Bath, UK (December 1991)
23. Pelachaud, C., Viaud, M.L., Yahia, H.: Rule-structured facial animation system. In: Proceedings of the 13th IJCAI, Chambery, France, pp. 1610–1615 (August 1993)
24. Essa, I.A.: Analysis, Interpretation and Synthesis of Facial Expressions. PhD thesis, MIT Media Lab (1994)
25. Saulnier, A., Viaud, M.L., Geldreich, D.: Real-time facial analysis and synthesis chain. In: Proc. Automatic Face and Gesture Recognition, Zurich (June 1995)
26. Konno, T., Mitani, H., Chiyokura, H., Tanaka, I.: Surgical simulation of facial paralysis. In: Sieburg, H.B., Weghorst, S.J., Morgan, K.S. (eds.) Medicine Meets Virtual Reality: Health Care in the Information Age, pp. 488–497. IOS Press, Amsterdam (1996)
27. Ekman, P., Friesen, W.: Facial Action Coding System. Consulting Psychologists Press, Inc., Palo Alto (1978)
28. Patterson, E.C., Litwinowicz, P.C., Greene, N.: Facial animation by spatial mapping. In: Computer Animation 1991, Tokyo. Springer, Heidelberg (1991)
29. Sturman, D.J.: Computer puppetry. Computer Graphics and Applications 18(1), 38–45 (1998)
30. Williams, L.: Performance-driven facial animation. In: Computer Graphics (SIGGRAPH 1990 Proceedings), vol. 24, pp. 235–242 (August 1990)
31. de Graf, B.: Performance' facial animation. In: SIGGRAPH 1989 Course Notes 22: State of the Art in Facial Animation, pp. 8–17 (July 1989)
32. Guenter, B., Grimm, C., Wood, D., Malvar, H., Pighin, F.: Making faces. In: SIGGRAPH 1998 (August 1998)
33. Lorenzo, M.S., James, D., Edge, S.A.K., Maddock, S.: Use and re-use of facial motion capture data. In: Vision, Video, and Graphics 2003, pp. 1–8, July 10-11(2003)

34. Deng, Z., Chiang, P.Y., Fox, P., Neumann, U.: Animating blendshape faces by cross-mapping motion capture data. In: Proc. of ACM SIGGGRAPH Symposium on Interactive 3D Graphics and Games, pp. 43–48 (2006)
35. Kshirsagar, S., Magnenat-Thalmann, N.: Visyllable based speech animation. Computer Graphics Forum 22(3), 631 (2003)
36. Cao, Y., Tien, W.C., Faloutsos, P., Pighin, F.: Expressive speech-driven facial animation. ACM Trans. Graph 24(4), 1283–1302 (2005)
37. Deng, Z., Neumann, U.: eFASE: expressive facial animation synthesis and editing with phoneme-isomap controls. In: Proceedings of SCA 2006, pp. 251–260. Eurographics Association (2006)
38. Deng, Z., Chiang, P.Y., Fox, P., Neumann, U.: Animating blendshape faces by cross-mapping motion capture data. In: I3D 2006: Proceedings of the 2006 symposium on Interactive 3D graphics and games, pp. 43–48. ACM, New York (2006)
39. Sifakis, E., Neverov, I., Fedkiw, R.: Automatic determination of facial muscle activations from sparse motion capture marker data. ACM Trans. Graph 24(3), 417–425 (2005)
40. Bickel, B., Botsch, M., Angst, R., Matusik, W., Otaduy, M., Pfister, H., Gross, M.: Multi-scale capture of facial geometry and motion. ACM Trans. Graph 26(3), 33 (2007)
41. Borshukov, G., et al.: Playable universal capture. In: ACM SIGGRAPH 2006 Sketches and Applications Program, Bostom, Ma (August 2006)
42. Ezzat, T., Poggio, T.: Videorealistic talking faces: A morphing approach. In: Proceedings of the ESCA Workshop on Audiovisual Speech Processing, Rhodes, Greece, pp. 141–144 (September 1997)
43. de Graf, B.: Performance' facial animation. In: SIGGRAPH 1990 Course Notes 26: State of the Art in Facial Animation, pp. 9–20 (August 1990)
44. Pelachaud, C.: Communication and Coarticulation in Facial Animation. PhD thesis, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, 19104-6389 (1991)
45. Ezzat, T., Poggio, T.: Miketalk: A talking facial display based on morphing visemes. In: Computer Animation 1998, pp. 96–102 (June 1998)
46. Gould, R.L.: Making 3-d computer character animation a great future of unlimited possibility or just tedious? In: SIGGRAPH 1989 Course Notes 4: 3-D Character Animation, pp. 31–63 (1989)
47. Kleiser, J.: A fast, efficient, accurate way to represent the human face. In: SIGGRAPH 1989 Course Notes 22: State of the Art in Facial Animation, pp. 36–40 (July 1989)
48. Pighin, F., Auslander, J., Lischinski, D., Salesin, D.: Realistic facial animation using image-based 3d morphing. Technical Report UW-CSE-97-01-03, University of Washington, Department of Computer Sceince & Engineering (1997)
49. Pighin, F., Hecker, J., Lischinski, D., ard Szeliski, R., Salesin, D.H.: Synthesizing realistic facial expressions from photographs. In: SIGGRAPH 1998, pp. 75–84 (1998)
50. Provine, J.A., Bruton, L.T.: Lip synchronization in 3-d model based coding for video-conferencing. In: Proc. of the IEEE Int. Symposium on Circuits and Systems, Seattle, pp. 453–456 (May 1995)
51. Waite, C.T.: The facial action control editor, face: A parametric facial expression editor for computer generated animation. Master's thesis, Massachusetts Institue of Technology (February 1989)
52. Morishima, S., Aizawa, K., Harashima, H.: A real-time facial action image synthesis system driven by speech and text. SPIE Visual Communications and Image Processing 1360, 1151–1157 (1990)

53. Waters, K., Levergood, T.M.: DECface: An automatic lip-synchronization algorithm for synthetic faces. Technical Report CRL 93/4, Digital Equipment Corporation Cambridge Research Lab (September 1993)
54. Platt, S.M., Badler, N.I.: Animating facial expressions. Computer Graphics (Proceedings of SIGGRAPH 1981) 15(3), 245–252 (1981)
55. Kähler, K., Haber, J., Seidel, H.P.: Geometry-based muscle modeling for facial animation. In: Graphics Interface 2001, 7–9 June 2001, pp. 37–46 (2001)
56. Koch, R.M., Gross, M.H., Carls, F.R., von Buren, D.F., Fankhauser, G., Parish, Y.I.: Simulating facial surgery using finite element models. In: Proceedings of SIGGRAPH 1996, pp. 421–428. Addison-Wesley, Reading (1996)
57. Koch, R.M., Gross, M.H., Bosshard, A.A.: Emotion editing using finite elements. Technical Report 281, Computer Science Department, ETH Zrich (1998)
58. Larrabee Jr., W.F.: A finite element model of skin deformation: Part III - the finite element model. Laryngoscope, 399–419 (April 1986)
59. Lee, Y., Terzopoulos, D., Waters, K.: Constructing physics-based facial models of individuals. In: Proceedings of Graphics Interface 1993, Toronto, Ontario, Canada, Canadian Information Processing Society, pp. 1–8 (May 1993)
60. Lee, Y.V.: The construction and animation of functional facial models from cylindrical range/reflectance data. Master's thesis, University of Toronto (1993)
61. Pieper, S.: Physically-based animation of facial tissue for surgical simulation. In: SIGGRAPH 1989 Course Notes 22: State of the Art in Facial Animation, pp. 71–124 (1989)
62. Rosenblum, R.E., Carlson, W.E., Tripp III, E.: Simulating the structure and dynamics of human hair: modelling, rendering and animation. Journal of Visualization and Computer Animation 2(4), 141–148 (1991)
63. Terzopoulos, D., Waters, K.: Physically-based facial modelling, analysis, and animation. JVCA 1(2), 73–80 (1990)
64. Viaud, M.L., Yahia, H.: Facial animation with wrinkles. In: Eurographics 1992, Cambridge, United Kingdom (September 1992)
65. Waters, K., Terzopoulos, D.: Modeling and animating faces using scanned data. The Journal of Visualization and Computer Animation 2(4), 123–128 (1991)
66. Wu, Y., Kalra, P., Magnenat-Thalmann, N.: Simulation of static and dynamic wrinkles of skin. In: Proceedings Computer Animation 1996, pp. 90–97 (1996)
67. Platt, S.M.: A Structural Model of the Human Face. PhD thesis, University of Pennsylvania, Philadelphia, Pennsylvania (1985)
68. Hofer, E.E.: A sculpting based solution for three-dimensional computer character facial animation. Master's thesis, The Ohio State University (1993)
69. Cassell, J., Vilhjálmsson, H.H., Bickmore, T.: BEAT: the behavior expression animation toolkit. In: Fiume, E. (ed.) Proceedings of SIGGRAPH 2001, pp. 477–486. ACM Press, New York (2001)
70. Pearce, A., Wyvill, B.M., Wyvill, G., Hill, D.: Speech and expression: A computer solution to face animation. In: Green, M. (ed.) Proceedings of Graphics Interface 1986, pp. 136–140 (May 1986)
71. Takashima, Y., Shimazu, H., Tomono, M.: Story driven animation. In: Carroll, J.M., Tanner, P.P. (eds.) Proceedings of Human Factors in Computing Systems and Graphics Interface 1987, pp. 149–153 (April 1987)
72. Magnenat-Thalmann, N., Thalmann, D. (eds.): Synthetic Actors in Computer-Generated 3D Films. Springer, Heidelberg (1987)
73. Fuchs, T., Haber, J., Seidel, H.P.: MIMIC – a language for specifying facial animations. In: WSCG SHORT Communication papers proceedings, pp. 71–78 (2004)

74. Cassell, J., et al.: Animated conversation: Rule–based generation of facial expression gesture and spoken intonation for multiple converstaional agents. In: SIGGRAPH 1994, 24–29 July 1994, pp. 413–420. ACM Press, New York (1994)
75. King, S.A., Knott, A., McCan, B.: Language-driven nonverbal communication in a bilingual conversational agent. In: CASA 2003, pp. 17–22 (May 7-9, 2003)
76. Lee, S.P., Badler, J.B., Badler, N.I.: Eyes alive. ACM Transactions on Graphics 21(3), 637–644 (2002)
77. Kent, R.D., Minifie, F.D.: Coarticulation in recent speech production models. Journal of Phonetics 5, 115–135 (1977)
78. Wang, A., Emmi, M., Faloutsos, P.: Assembling an expressive facial animation system. In: Sandbox 2007: Proceedings of the 2007 ACM SIGGRAPH symposium on Video games, pp. 21–26. ACM, New York (2007)
79. Löfqvist, A.: Speech as audible gestures. In: Hardcastle, W.J., Marchal, A. (eds.) Speech Production and Speech Modeling, pp. 289–322. Kluwer Academic Publishers, Dordrecht (1990)
80. Cohen, M., Massaro, D.: Modeling coarticulation in synthetic visual speech. In: Models and Techniques in Computer Animation, pp. 139–156. Springer, Heidelberg (1993)
81. Le Goff, B.: Automatic modeling of coarticulation in text-to-visual speech synthesis. In: Proccedings of Eurospeech 1997, Rhodes, Greece, vol. 3, pp. 1667–1670 (September 1997)
82. Albrecht, I., Haber, J., Seidel, H.P.: Speech synchronization for physics-based facial animation. In: Proceedings of WSCG 2002, 4-8 February 2002, pp. 9–16 (2002)
83. Kähler, K., Haber, J., Yamauchi, H., Seidel, H.P.: Head shop: Generating animated head models with anatomical structure. In: SCA 2002, 21-22 July 2002, pp. 55–64 (2002)
84. Albrecht, I., Haber, J., Seidel, H.P.: Automatic generation of non-verbal facial expressions from speech. In: Proceeding of CGI 2002, 3–5 July 2002, pp. 283–293 (2002)
85. Kim, T.Y., Neumann, U.: Interactive multiresolution hair modeling and editing. ACM Transactions on Graphics 21(3), 620–629 (2002); (Proceedings of SIGGRAPH 2002)
86. Marschner, S., Jensen, H.W., Cammarano, M., Worley, S., Hanrahan, P.: Light scattering from human hair fibers. ACM Transactions on Graphics 22(3), 780–791 (2003); (Proceedings of SIGGRAPH 2003)
87. Bando, Y., Chen, B.Y., Nishita, T.: Animating hair with loosely connected particles. Computer Graphics Forum 22(3), 411 (2003)
88. Koster, M., Haber, J., Seidel, H.P.: Real-time rendering of human hair using programmable graphics hardware. In: Proc of CGI 2004 (June 2004)
89. Haro, A., Guenter, B., Essa, I.: Real-time, photo-realistic, physically based rendering of fine scale human skin structure. In: Rendering Techniques, pp. 53–62 (2001)
90. Jensen, H.W., Marschner, S.R., Levoy, M., Hanrahan, P.: A practical model for subsurface light transport. In: Fiume, E. (ed.) Proceedings of SIGGRAPH 2001, August 2001, pp. 511–518. ACM Press, New York (2001)
91. Marschner, S.R., Guenter, B.K., Raghupathy, S.: Modeling and rendering for realistic facial animation. In: Rendering Techniques 2000, pp. 231–242 (2000)
92. Weyrich, T., et al.: Analysis of human faces using a measurement-based skin reflectance model. In: SIGGRAPH '06: ACM SIGGRAPH 2006 Papers, pp. 1013–1024. ACM, New York (2006)

Autonomous Digital Actors

Ken Perlin[1] and Gerry Seidman[2]

[1] NoiseMachine, New York University, New York, NY, USA
[2] Actor Machine, LLC
`perlin@noisemachine.com, seidman@actorMachine.com`

Abstract. Using Procedural Animation techniques, Autonomous Digital Actors can be created that can take blocking and emotive direction to act out scenes without the use of linear or motion captured animation. We describe the high level architecture of our research that enables (1) interfaces that "coach" the movement styles of digital actors, and (2) the incorporation of pre-existing motion captured and hand animated walk cycles in this coaching process.

Keywords: Animation, procedural, autonomous.

1 Introduction

The future of computer games and animated films is not in spaceships or dragons or laser death rays, but rather in the mysteries of the human heart. What makes a relationship really work? What is the magic within human bonds of friendship, sexual chemistry, the intuitive understanding of why somebody has just told you the exact opposite of what they meant, and yet communicated to you the deeper truth beneath? We are moved by great performances, either live or animated, by the expression of the inner thoughts and emotions of the characters by the talent of the actor or animator. Translating these concepts to real-time interactive animation requires moving beyond mere blending of pre-existing snippets of linear animation.

The results of recent research suggests that much of the 3D animation and gaming industry will soon be shifting to a new way to create and animate 3D characters, and that rather than being required to animate a character separately for each motion sequence, animators will be able to interact with software authoring tools that will let them train an *Autonomous Digital Actor* (ADA) how to employ various styles of movement, body language, techniques for conveying specific emotions, best acting choices, and other general performance skills.

Once properly trained, such an ADA would be able to take direction interactively from a non-animator, to play many different scenes while effectively conveying changing nuances of mood, personality and intention.

Directors may become accustomed to working with troupes of ADAs that can interactively play scenes together, and are able to follow stage directions and sight

Lines, manipulate props, and effectively express the shifting relationships between characters in the scene.

© Springer-Verlag Berlin Heidelberg 2008

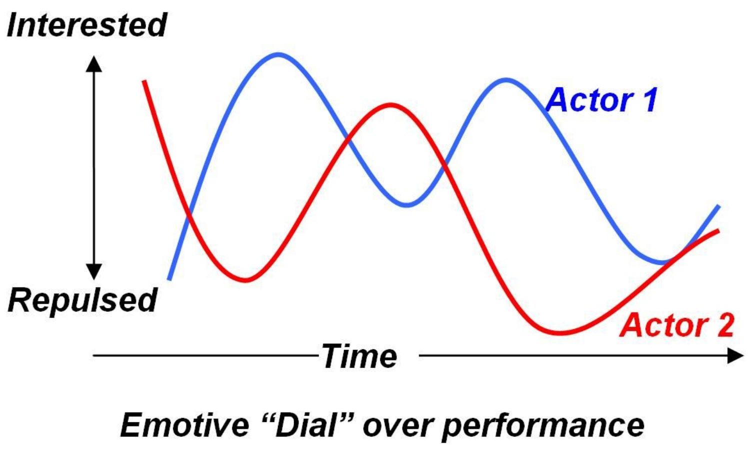


Fig. 1. Emotive Dial over Performance

Such capabilities can lead to new styles of production practice that have the potential to shift the economics of character animation. The ability to impart skills to an ADA would also amount to a new form of intellectual property. Animators who teach such skills, or the employers who pay those animators' salaries, would be able to sell and trade such I.P. to anyone who wishes to make animated movies and interactive entertainment.

Another consequence of such developments could be an eventual convergence between animated movies and interactive media, both in their production practices and in their content, potentially evolving into new genres of interactive character-driven storytelling that can emotionally engage audiences in ways that cannot be achieved by either movies or games.

The increased use of ADA's may also lead to an explosion in user-created animated content, which could have a transformative impact on public venues for the distribution of user-created content such as *YouTube*.

2 Our Research Approach

This paper describes the high level architecture of an approach we have been taking toward the development of ADAs, in which animators explicitly take on the role of an acting coach. We will refer to the software used by such animators as *Acting Coach*.

Using procedural animation techniques, movement, gestures, body language, as well as emotional relationships are expressed parametrically and then simulated at run-time rather than stored as animation clips. Unlike hand animation, procedural animations assets can be 'dialed up/down', to control their relative contributions to the final movement, as well as combined with any other procedural asset.

Integrating this approach into existing game run-time engines is straightforward, as our procedural animation tools are responsible only for providing movement of bone matrices.

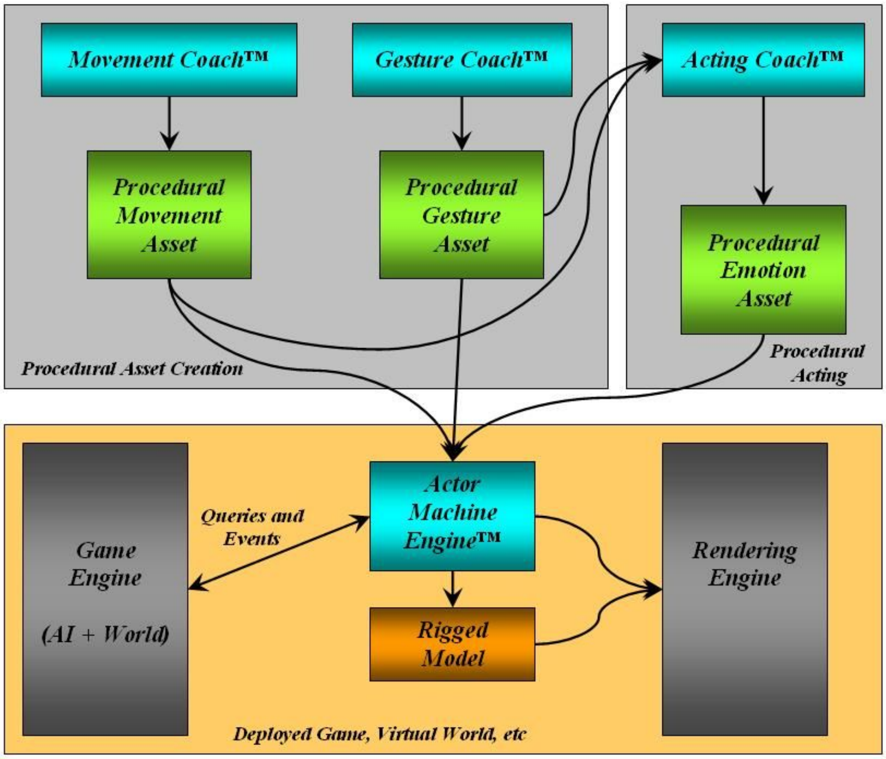


Fig. 2. The Software Stack

Within an application, the Actor Machine Engine views the AI through physicality along the following layering

- Response to other actors
  (Example: Get angry when your date checks out another character)
- Emotional Direction
  (Example: Act Angry, confused, wistful, weary)
- Physical Direction
  (Example: Shoulders back, knees bent)
- Physicality
  (Example: Joint movement)

The top layer would be part of a game AI, director's authoring tool (whether text based or through a GUI) while the bottom three layers comprise the Actor Machine Core layering.
Types of Input to the Actor Machine Engine:

- Emotional direction for each character
- Inter-Character Relationship
- Character Blocking Goals (including such actions as 'meander')

- Character Action Goals (pick something up, sit down, embrace another character, etc)
- Terrain
- Obstacles
- Props (augmented with, for example, grasping handles)

The Engine can be queried and/or events generated about its current state for interaction with the Game AI (such as goals reached, objects grasped, etc).

Since the procedural engine produces typical bone matrices for animation pipeline, our animations can be blended in/out with linear animation clips or ragdoll animation using traditional techniques.

By simplifying the interaction with the animation subsystem, we hope to show that a procedural character engine can become an easy to integrate part of the game stack, much as physics engines have become.

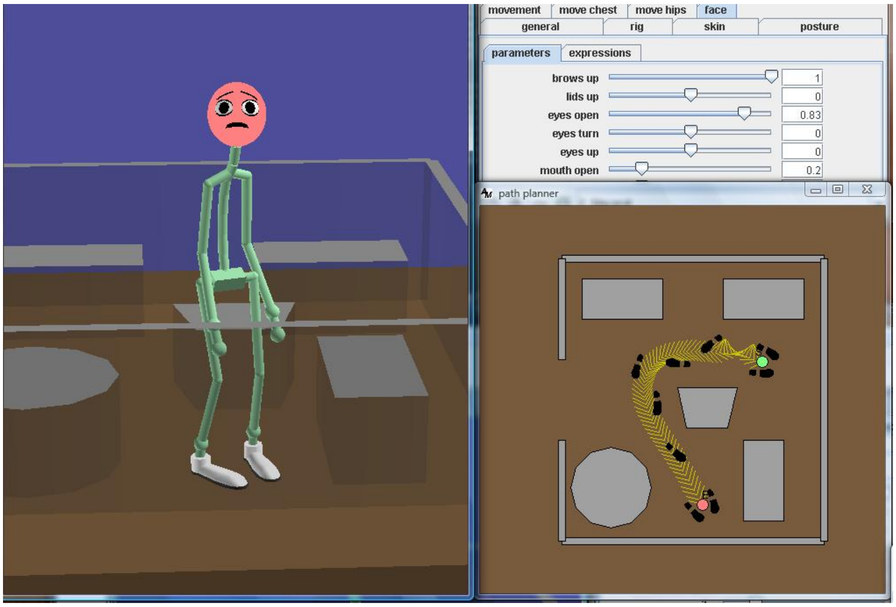
3 The Components

Authoring tools are required to train digital actors in the form of *Procedural Assets*, while run-time engines are required to generate the movement based on these assets. We refer to authoring tools as *Coaches* and the runtime engines as *Machines*. In this section we delineate the various types of Coaches and Machines. It is important to note that the various Coaches are aimed at different types of users. For example, the *Movement Coach* would be used by a trained animator who understands the subtlety of human movement. The *Acting Coach* would not require animation skills. It is used to train the digital actors how to utilize gestures and movements available to it (as assets created by animators) to express coherent emotional states and actions.

3.1 Movement Coach

The *Movement Coach* is used to train a digital actor as to the basic aspects of how to walk, move, sways, rhythms, etc. by creating/editing *Procedural Movement Assets*. In essence, an animator would use this tool to tune the 'style' of movement for that digital actor (eg: John Wayne, Marilyn Monroe and Shrek each have their own unique style of movement). A core set of base parameters are provided. These parameters can be used in combination to create complex and unique movement styles.

Creation of a walk style primarily involves the rhythmic relationships between parts of the actor's movement. Aspects of walking, such as turning, stopping/starting, etc can be simulated based on the body proportions/mass and the rhythm of the movement. Once a walk style has been created, procedural actors can automatically walk along any path, terrain (even moving terrain), start/stop, stand/sit, walk up/down stairs, etc. without anything more than these procedural movement assets. The assets created with the *Movement Coach* are core building blocks used in the other modules described below.



brows up	1
lids up	0
eyes open	0.83
eyes turn	0
eyes up	0
mouth open	0.2

Fig. 3. Movement Coach

3.2 Walking Machine / Blocking Machine

Utilizing procedural movement assets and an arbitrary path (over arbitrary terrain), the *Walking Machine* synthesizes appropriate foot placement and body movement along that path. Because of the nature of the *Procedural Movement Assets*, the walk styles may be blended over time producing emotive transformations. For example, an actor may be directed to change their style from happy to sad as they walk across the room (for example coinciding with another actor telling them bad/sad news).

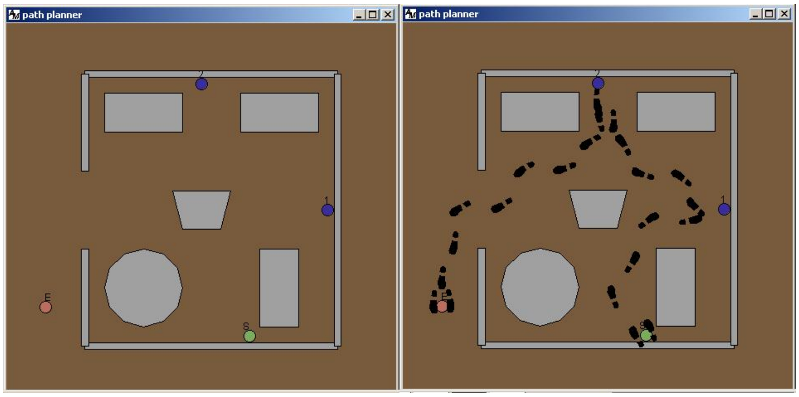


Fig. 4. Path Planning

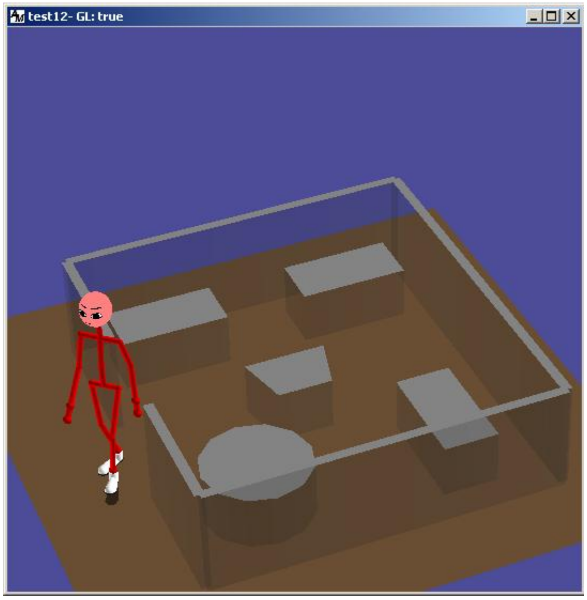
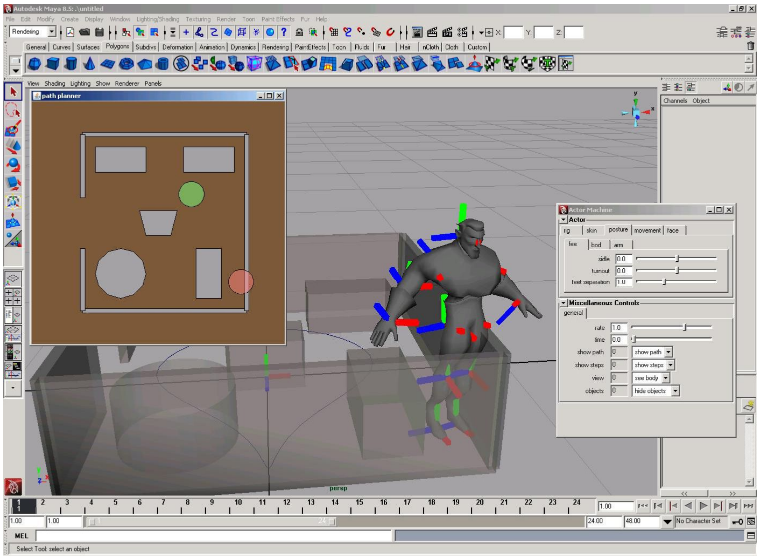


Fig. 5. Actor Walking


Autodesk Maya 8.5: .\untitled
File Edit Modify Create Display Window Lighting/Shading Texturing Render Toon Paint Effects Fur Help
Rendering
General Curves Surfaces Polygons Subdivs Deformation Animation Dynamics Rendering PaintEffects Toon Fluids Fur Hair nCloth Cloth Custom
View Shading Lighting Show Renderer Panels
path planner
Channels Object
persp
MEL
Select Tool: select an object
No Character Set

Actor Machine
Actor
rig | skin | posture | movement | face
fee | bod | arm
sidle 0.0
turnout 0.0
feet separation 1.0
Miscellaneous Controls
general
rate 1.0
time 0.0
show path 0 show path
show steps 0 show steps
view 0 see body
objects 0 hide objects

Fig. 6. The Blocking Coach/Engine as a Maya Plug-In

The *Blocking Machine* is situated above the *Walking Machine* allowing the user (or game AI) to set up waypoints for one or more characters in a scene to hit as they perform. Using *Procedural Movement Assets*, the *Path Planner* will find an appropriate path given the obstacles in the scene. The *Walking Machine* takes on the responsibility of correctly placing the feet, moving the head, hips and shoulders, and correcting for arm swing, etc., all while maintaining the designated emotional movement style and cadence of the actor. While the *Blocking Machine* choreographs the timing and blocking of the scene, the *Walking Machine* guarantees correct physicality of the performance. Depending on the goals presented to the *Blocking Machine*, it will achieve them while maintaining sight lines based on the heights of the obstacles and gazing targets.

There are many types of 'emotional dials' that can control the emotive movement of the actor. For example the user could 'turn up or down how tired, sad, happy, arrogant, wounded/limping (and in combinations) a character is. These emotional dials can be curves over time changing as the character moves through the scene.

To encourage the creation of procedural assets as well as experimentation with authoring procedural animation scenes, we have wrapped the *Blocking Machine* in both a *Maya* Plug-in and *3DS Max* Plug-In.

3.3 Gesture Coach

The *Gesture Coach* is used to train a digital actor about the basic hand and facial gestures by creating/editing *Procedural Gesture Assets.* These gestural assets may represent such actions as pointing, hand waving, chin stroking, leers, smirks, etc.

Unlike with blended animation techniques, these gestures can shaded by other directable properties of the actor's movement. Because of the nature of *Procedural Gesture Assets*, they can be sharpened/lessened (intensity), made wider or tighter (expansiveness), and directed towards a fixed of moving target. Additionally they can be used in any combinations, for example a wide arm swing gesture can be used along with a sharp dismissive hand gesture creating a complex hand effect.

3.4 Acting Coach

The *Acting Coach* is where emotive training occurs. While *Movement and Gesture Assets* are lower level assets, defining the vocabulary of movements that are available to a procedural actor, *the Acting Couch* is used to creating/editing *Procedural Emotion Assets.*

With the *Acting Coach* the user defines emotive styles by combining the lower level gestures. For the training of a particular actors way of conveying "nervous" would entail certain movement styles and certain gesture combinations that are available to the actor for this emotive state. The actor will chose from the vocabulary of gestures to perform either in a somewhat random manner or based on queues and triggers. *Emotional Assets* can be thought of as weighted chords of notes of Movement/Gestural assets.

High level Emotional Assets allow for complex character animation with simple direction rather than animation. The director, either interactively in a desktop application or controlled by a game AI, has access to a many types of 'emotional dials' that they might choose to use, for example the director can turn up or down how tired, sad, happy or arrogant (or combinations) a character is. Similarly the director can control the level and type of emotional relationships in a scene between the characters such as their respective mutual interested, fear or sexual tension. Of course these emotive moods and relationships will change over time in a scene to reflect the 'inner thoughts' of the characters. These directions would affect the body language, glancing, hand gestures, facial expressions or other appropriate 'natural' expressions of the direction. How the character expresses these emotions are based on how the digital actor was trained to act. Unlike blended animation, all motion is procedurally generated, so the performance need not convey the 'sameness' of motion that is endemic to linear animation-based techniques often used in current games and virtual worlds. Too much sameness in a character's movements can break the audience's belief that the character is 'alive'.

These *Procedural Emotion* Assets are used along with blocking information and character relationships by the Actor Machine Core Engine to produce believable characters as they move through a scene while expressing emotional and relational shifts. Directors can plan performances based on the emotional curve that occurs during a scene. For example, a scene might contain two characters, one of whom is becoming progressively more attracted to the other. Authoring tools and engines based on procedural animation techniques can allow the virtual actors to automatically convey such shifts (drawing on the "coaching" that has already been incorporated into them), thereby enabling a director to simply draw emotional curves describing the evolving emotional states/relationships of the characters over time.

3.5 The Proceduralizer

Through our research we were able synthesize *Procedural Assets* by analyzing *Motion Capture* and/or Hand animation to obtain high-fidelity procedurally synthesized animation. We have implemented this in a tool we refer to as the *Proceduralizer*. For many animation studios and game development companies, it is critical to provide a path that allows them to capitalize on their existing database of linear animations by transforming those assets into procedural assets.

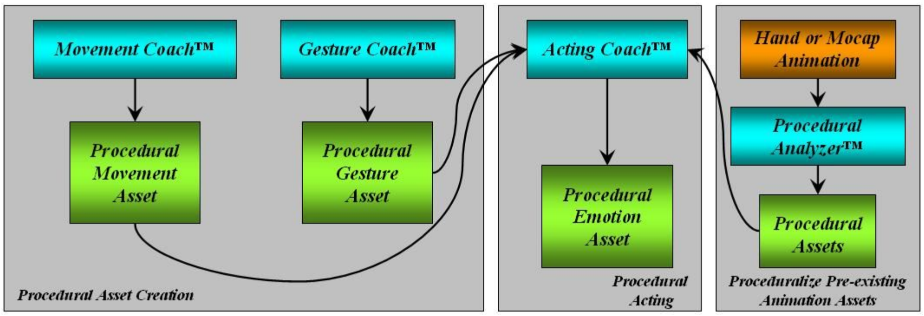


Fig. 7. Converting Hand/Mocap Data into Procedural Data

The Proceduralizer can, for example, transform a single walk cycle of data into a *Procedural Movement Asset*. This could then be used with the *Blocking Machine* to generate walking along any path. The animation produced retains the fidelity of movement from that single (Mocap or Hand Animated) cycle and be able to apply it to arbitrary movement throughout the scene including starting/stopping, turning, walking on uneven terrain, and up/down stairs.

4 Conclusion

We have described the high level architecture of an approach to procedural animation that allows digital actors to perform in real-time while conveying directable nuances of emotion and intention. Given the limitations of space, our aim for this paper was not to convey the low level details of this system, but rather the philosophy we have taken to the design of the system, and high level relationships between its software components. Our hope is that the availability of systems incorporating such procedural approaches to digital actor movement will drive the movement of real-time digital actors away from mere blending of pre-existing animations, and toward true directed performance.

References

1. Amaya, K., Bruderlin, A.: Emotion from motion. In: Proceedings of the Conference on Graphics Interface 1996, Toronto, Canada (1996)
2. Assanie, M.: Directable synthetic characters. In: Proceedings of the 2002 AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment (2002)
3. Bates, J., Loyall, A., Reilly, W.: An architecture for action, emotion, and social behavior. In: Castelfranchi, C., Werner, E. (eds.) MAAMAW 1992. LNCS, vol. 830, pp. 55–68. Springer, Heidelberg (1994)
4. Ekman, P.: An argument for basic emotions. In: Stein, N., Oatley, K. (eds.) Basic Emotions, pp. 169–200. Lawrence Erlbaum, Hove (1992)
5. Hodgins, J.K., Pollard, N.S.: Adapting simulated behaviors for new characters. In: Proceedings of ACM SIGGRAPH 1997 Conference (1997)
6. Isla, D., Burke, R., Downie, M., Blumberg, B.: A layered brain architecture for synthetic creatures. In: Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI), Seattle, WA (2001)
7. Liu, C.K., Popovic, Z.: Synthesis of complex dynamic character motion from simple animations. ACM Trans. Graphics 21 (2002)
8. Murray, J.: Hamlet on the Holodeck: The Future of Narrative in Cyberspace. MIT Press, Cambridge (1998)
9. Ortony, A.: On making believable emotional agents believable. In: Trappl, R., et al. (eds.) Emotions in Humans and Artifacts. MIT Press, Cambridge (1988)
10. Perlin, P.K., Goldberg, A.: Improv: A system scripting interactive actors in virtual worlds. In: Proceedings of the ACM SIGGRAPH 1996 Conference, New Orleans, LA (1996)
11. Perlin, P.: Layered Compositing of Facial Expression. In: ACM SIGGRAPH 1997 Technical Sketch (1997)
12. Perlin, P.: Real Time Responsive Animation with Personality. EEE Transactions on Visualization and Computer Graphics 1(1) (1994)
13. Reilly, W.S.N.: Believable social and emotional agents. Ph.D dissertation, School of Computer Science, Carnegie Mellon Univ., Pittsburgh, PA (1996)

14. Rose, C.: Verbs and adverbs: Multidimensional motion interpolation using radial basis functions. Ph.D. Dissertation, Dept. of Computer Science, Princeton Univ., Princeton, NJ (1999)
15. Schaub, H., Zoll, C., Aylett, R.: Modelling empathy: The EU-project VICTEC (virtual information and communication technology with empathic characters). In: Proceedings of the Fifth International Conference on Cognitive Modeling, Bamberg, Germany (2003)
16. Steed, P.: Animating Real-Time Game Characters. Charles River Media, Hingham (2002)
17. Tomlinson, B.: From Linear to Interactive Animation: How Autonomous Characters Change the Process and Product of Animating University of California, Irvine
18. Tomlinson, B., Blumberg, B., Nain, D.: Expressive autonomous cinematography for interactive virtual environments. In: Proceedings of the 4th International Conference on Autonomous Agents, Barcelona, Spain (2000)

Author Index