SIGRAD 2008

The Annual SIGRAD Conference

Special Theme: Interaction

November 27–28, 2008

Stockholm, Sweden

Conference Proceedings

Organized by

SIGRAD, Swedish Chapter of Eurographics

and

Royal Institute of Technology

Edited by

Kai-Mikael Jää-Aro & Lars Kjelldahl

Published for

SIGRAD, Swedish Chapter of Eurographics

by Linköping University Electronic Press

Linköping, Sweden, 2008

Linköping Electronic Conference Proceedings, No. 34
Linköping University Electronic Press
Linköping, Sweden, 2008

ISBN 978-91-7393-733-7
ISSN 1650-3686 (print)
http://www.ep.liu.se/ecp/034/
ISSN 1650-3740 (online)
Print: LiU-Tryck, Linköpings universitet, 2008

Cover illustrations selected from the papers.

Table of Contents

Preface

We are happy to present a very interesting program for the annual SIGRAD conference, continuing the trend of international participation. The theme of the conference this year is “Interaction” and thus we are very pleased to cooperate with STIMDI, the Swedish human-computer interaction society.

The Royal Institute of Technology (KTH), hosts this year’s conference 27–28 November 2008. One aim has been to raise the awareness of Eurographics in Sweden in preparation for Eurographics 2010, which will be hosted by the Norrköping Visualization and Interaction Studio.

We thank the main sponsor of this event, VIC (Visualization, Interaction, and Communication), the network for visualization at KTH.

We thank the international program committee for their support and their valuable review work. Finally, thanks to all the authors and the conference participants without whom this conference would not have been possible.

We wish all participants a stimulating conference, and hope they take the chance to get involved in the Swedish and European graphics community.

Welcome to the conference of the Swedish Chapter of Eurographics, SIGRAD 2008!

Kai-Mikael Jää-Aro   Lars Kjelldahl

Editors

Conference organizing committee

Nils-Erik Gustafsson, IT-Arkitekterna
Anders Hast, University of Gävle
Kai-Mikael Jää-Aro, Telestream AB
Lars Kjelldahl, Royal Institute of Technology
Thomas Larsson, Mälardalen University

International program committee

Ewert Bengtsson, Uppsala University
Matt Cooper, Linköping University
Modris Dobelis, Riga Technical University
Nils-Erik Gustafsson, IT-Arkitekterna
Anders Hast, University of Gävle
Kai-Mikael Jää-Aro, Telestream AB
Lars Kjelldahl, Royal Institute of Technology
Ivana Kolingerová, University of West Bohemia in Pilsen
Ann Lantz, Royal Institute of Technology
Thomas Larsson, Mälardalen University
Lennart Ohlsson, Lund University
Yngve Sundblad, Royal Institute of Technology
Gustav Taxén, Avalanche
Anders Ynnerman, Linköping University

Keynotes

Human-Computer Interaction for Visualisation

Yngve Sundblad
Royal Institute of Technology

Abstract

In many cases visualised phenomena can be better understood using interactive devices and means for navigation in space and time, highlighting, different viewpoints and representations, recalculation and revisualisation of interesting parts, etc.

In the presentation some examples will be given on how HCI knowledge can be used for such situations in visualisations in the broad sense, perceptualisations with several senses (visual, audial and haptic). Initiatives for forming visualisation networks in Stockholm will also be mentioned.

Gapminder

Unveiling the beauty of statistics for a fact based world view

Flash animations bringing enormous amounts of data to life

Daniel Lapidus
Gapminder

Abstract

Gapminder is a non-profit venture promoting sustainable global development and achievement of the United Nations Millennium Development Goals by increased use and understanding of statistics and other information about social, economic and environmental development at local, national and global levels.

Gapminder developed the Trendalyzer software, which converts boring numbers into enjoyable, animated and interactive graphics–thus unveiling the beauty of statistical time series.

In March 2006, Google acquired Trendalyzer from Gapminder Foundation and the team of developers working for Gapminder joined Google in California in April 2007.

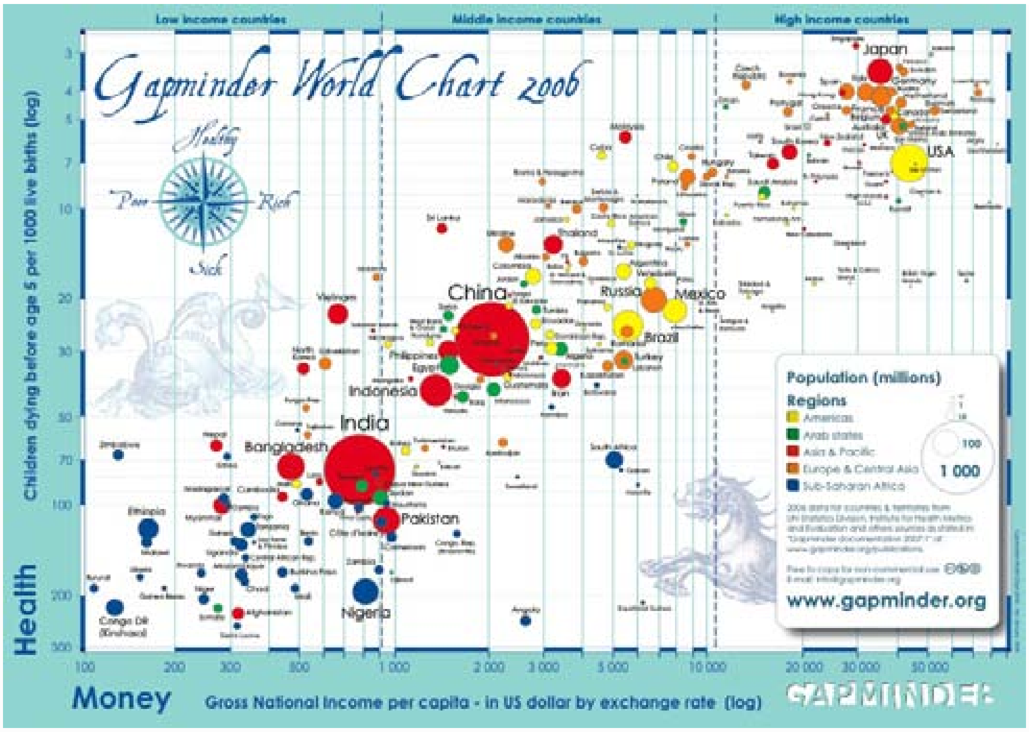
Since the Trendalyzer development was taken over by Google, the Gapminder Foundation maintain the same aim and uses Trendalyzer and its resources to produce videos and web services showing major global development trends with animated statistics.

The time series are made freely available in the web service called Gapminder World (www.gapminder.org/world/). Gapminder World enables the end users to further explore the underlying statistics in Trendalyzer graphics.

This seminar will focus on:

- Dissecting Trendalyzer (technically oriented)
- Gapminder & Trendalyzer in the future

Keywords: Gapminder, Trendalyzer, Flash/Flex for visualizations, Google APIs, usability



Experiential qualities of interactive visualizations

Jonas Löwgren
Malmö University

Abstract

A key factor in designing interactive visualizations is what makes them good to use and, more generally, what "good to use" means in the context of interactive visualizations.

In this talk, I introduce the notion of experiential qualities as a way to articulate what "good to use" means for a specific design domain. I then look specifically at interactive visualizations, arguing that the qualities of pliability and meaning are particularly important for the user and hence for the designer.

The case for pliability and meaning is built around a number of contemporary design examples, mainly drawn from interactive information visualization.

State of the art report

Haptics

Haptic Force Feedback in Mediated Interaction

Eva-Lotta Sallnäs
Royal Institute of Technology

Abstract

In face-to-face communication and collaboration people are used to being able to both see and hear other persons. People also take for granted the possibility to give objects to each other, to shake hands or to get someone's attention by a pat on the shoulder. However, most systems for mediated collaboration do not take physicality into account. Now emerging media space technologies like three-dimensional haptic interfaces makes it possible to interact physically in shared haptic object spaces. Many questions then arise about the effects of these modalities on communication and collaboration.

1 The sense of touch

The perception of touch is complicated in nature. The human touch system consists of various skin receptors, receptors connected to muscles and tendons, nerve fibres that transmit the touch signals to the touch centre of the brain, as well as the control system for moving the body. Different receptors are sensitive to different types of stimuli. Tactile perception is defined as perception mediated solely by variations in cutaneous stimulation (Loomis and Lederman, 1986). There are receptors sensitive to pressure, stretch of skin, location, vibration, temperature and pain. Contrary to what one might think, there does not seem to be one receptor type for sensing pressure, another for sensing vibration and so forth. Rather, the different receptors react to more than one stimulus type (Burdea, 1996). The skin on different parts of the body is differentially sensitive to touch. The ability to localise stimulation on the skin depends on the density of the receptors, which are especially dense in the hands and face. Moreover, a great deal of information provided by the kinesthetic system is used for force and motor control. The kinesthetic system enables force control and the control of body postures and motion. The kinesthetic system is closely linked with the proprioceptic system, which gives us the ability to sense the position of our body and limbs. Kinesthetic perception is defined as perception from joints and muscles, by limb movement alone, of hardness, viscosity and shape (Loomis and Lederman, 1986). Receptors (Ruffini and Pacinian corpuscles, and free nerve endings) connected to muscles and tendons provide the positional information. Haptic sensing is defined as the use of motor behaviours in combination with touch to identify objects (Appelle, 1991). In haptic perception both the cutaneous sense and kinesthesis convey significant information about distal objects and events. The haptic system unifies input from many sources, e.g., position of fingers, pressure, into a unitary experience.

Manipulation of objects can take many forms and one taxonomy illustrates how diverse functions haptics fulfils in everyday life (Lederman, and Klatzky, 1987). People use different strategies depending on the purpose of the tactile manipulation, such as investigating the weight, form, texture or softness of an object. Joint manipulation of objects can take just as many forms. One example is jointly grasping an object and moving it through an area that might have restrictions (Ruddle et al., 2002). Another example is moving an object by pushing from both sides and lifting the object together. Yet another type of joint manipulation is grasping an object and handing it to another person. In a shared haptic object-space people can coordinate joint movement of objects by signalling direction through haptic force and they can give objects to each other almost without verbal communication. Also, the bilateral (Biggs and Srinivasan, 2002) qualities of haptic perception make it possible to both move an object and to get information from it at the same time. In a collaborative situation these aspects of haptic sensing might facilitate the joint understanding of complex information or how something is constructed

2 Psychology of touch

The use of the sense of touch for understanding information in the form of texture and shape is often neglected in computer interface design because of the traditionally perceived dominance of vision for interacting with graphical objects. Touch has by a number of philosophers been seen as dominant over other senses in terms of an existence proof for objects, that is, we test reality of a mirage or illusion by trying to touch it (Heller and Schiff, 1991). Humans tend to think of touch as the "reality sense" because we know that it is relatively easy to fool vision by distorting lenses, differences in lightning and viewing conditions. Traditionally touch has been dismissed as a lower sense whereas vision and hearing are looked upon as the higher senses. Katz (1989) however, argued that touch from a perceptual viewpoint must be given precedence over all other senses because its perceptions have the most compelling character of reality. Katz argued that:

"touch plays a far greater role than do the other senses in the development of belief in the reality of the external world. What has been touched is the true reality that leads to perception; no reality pertains to the mirrored image, the mirage that applies itself to the eye."

Other senses are more ambiguous than touch and therefore touch is often used to check on reality. It is hard to imagine that we would believe what we see rather than what we feel. Most people think that an object is rather stable over time regarding its size and shape. This is probably a consequence of the fact that, even though the retinal size and shape of an object can differ due to viewing conditions, angles and distance, the touch percept is more or less stable. We think that an object has only one true size and shape and only one true surface structure. The fact that people generally perceive touch percepts to be stable becomes a very important aspect to consider when designing haptic interfaces. Because people trust the haptic perception, and are usually not used to simulated haptics, inconsistencies in the haptic simulation can have serious consequences. One problem is that people explore what they see to a larger extent than things that are invisible but haptically perceivable. This means that, if great care is taken to design a complete haptic object, but the visual graphics for example only reveal parts of the haptic model of the object, the risk is large that only the visible parts will be explored. Some haptic illusions can also surface because perceptual events that are very infrequent in the real world can be easily simulated. One example is that if two boxes with different sizes but equal weight, that seem to be of the same material, are lifted by a person, the larger object is perceived to be lighter than the smaller one. This is because in nature a larger object should be heavier than a smaller one if they are of the same material.

Gibson (1979) argues that all aspects of the world provide affordances. Ground affords support for walking, air affords breathing, water affords drinking and solid materials afford manipulation by the human body and primarily the hands. Depending on the qualities that a solid material has it affords different kinds of manipulation and different things can be manufactured, usually fabricated by hand. Gibson (1979) argued that:

"To identify the substance in such cases is to perceive what can be done with it, what it is good for, its utility; and the hands are involved"

Gibson (1979) gives a number of examples of affordances that different objects have that depend on their properties or qualities: colour, texture, composition, size, shape and features of shape, mass, elasticity, rigidity and mobility. An elongated object of moderate size affords wielding, hitting, or raking. A graspable rigid object affords throwing and an elongated elastic object affords binding or weaving. In contrast to many other psychologists Gibson thought that phenomenal (psychologically perceived) objects are not built up of easily discriminative parts or qualities but are instead perceived as integrated unified entities that afford certain behaviours. We identify an object as one whole entity, one specific thing, not as a bunch of separate qualities. Gibson (1979) argued that:

"It is never necessary to distinguish all the features of an object and, in fact it would be impossible to do so. Perception is economical. Those features of a thing are noticed which distinguish it from other things that it is not - but not all the features that distinguish it from everything that it is not"

Among researchers that study the tactile sense, the importance of movement in relation to touch perception has been recognised (Gibson, 1979; Katz, 1989). Gibson thought that movement was essential for perception, the movement of the limbs and head relative to the body and the locomotion relative to the environment. Accordingly, Gibson makes a distinction between passive and active touch. Touch is passive when the person does not move and information is imposed on the skin (Heller and Schiff, 1991). Active touch consists of self-produced movement that allows the perceiver to obtain objective information about the world. It was shown that people rely a lot on explorative movement to recognise shapes when blindfolded. In an experiment it was found that when "cookie cutter" shapes were pressed into the palm of the hand of the subject (passive touch) the shape recognition was as low as 29% whereas recognition was 95% when subjects could explore (active touch) the shape freely (Appelle, 1991). In essence it is generally argued that haptic perception is active touch as information is obtained through both tactile perception by the nerves in the skin and kinestetic perception by nerves in the muscles and joints. In the use of haptic interfaces, touch is usually active rather than passive.

Gibson (1979) argued that humans not only perceive the affordances of objects but that also the social behaviour of other beings, including animals, have affordances. Humans are dynamic and convey complex patterns of behaviour that other humans interpret as affording certain behaviours reciprocally in a continuous fashion. Humans interact with one another and behaviour affords behaviour. Nurturing behaviour, fighting behaviour, cooperative behaviour, economic behaviour, political behaviour – all depend on the perceiving of what another person affords, or sometimes the misperceiving of it. Gibson (1979) argued that:

"The perceiving of these mutual affordances is enormously complex, but nonetheless lawful, and it is based on the pickup of the information in touch, sound, odour, taste and ambient light"

Following this line of reasoning, multimodal input of information is important for an accurate understanding of another person's social affordances. In mediated interaction only a selection of a person's affordances in the real world can be conveyed to a receiver. In for example text-only communication a person can only communicate the message in text and the receiver of the message has to imagine for example emotions through the text description. Another person's voice conveys much more detailed social affordances through pitch, loudness, tempo and melody. Video conveys even more communication behaviour that hypothetically would improve social affordances but video is sometimes more unreliable than audio as the optic system can distort many of the social signals such as eye gaze, body size and distance.

Haptic feedback is taken for granted in our non-mediated interaction with others. Tactile contact with others is managed in very subtle ways because of the fundamental importance of protecting ourselves from harm at the same time as tactile contact is probably essential for well-being and survival. Gibson (1979) includes the importance of all senses for perceiving social affordances of others as well as for perceiving affordances of objects around us. Mehrabian (1972) includes touching as the most important variable in his construct “immediacy” along with interpersonal distance, forward leaning toward the addressee, eye contact and body orientation in that order of importance. Immediacy is one of the concepts that influenced Short et al. (1976) in their work on the social presence theory. However, haptic feedback is just starting to be used in interface design for interaction with graphical objects and has been used even less for mediated human interaction and joint manipulation of objects. Tactile contact in real encounters provides the most fundamental proof of something being real and believable. This is probably also true in social interaction even when the haptic interaction is very limited, as when two divers pull at each end of a rope, a buddy-line, when diving in murky waters in order to stay in contact. The tactile contact is an important aspect of social interaction but even more so might the tactile contact be that people avoid, like hitting the other on the nose. These aspects are most probably important when for example building trust between people.

3 Haptic shared virtual environments

In the real world, haptics is frequently involved in human-human interaction, like hand shaking or tapping someone on the shoulder. Handing over objects is for example a common event in face-to-face interaction. A frequent and watchful example of this occurs when being given a cup of coffee in an airplane both the flight attendant and the customer have to pay attention to subtle haptic signals to ensure that the hand off is securely accomplished. The question is how such an event can be supported when the interaction takes place in a shared virtual environment. Although not as well studied as single user interface interaction, a few authors have investigated issues regarding joint manipulation of virtual objects in a haptic collaborative virtual environment (Ishii et al., 1994; Basdogan et al., 2000; Sallnäs et al., 2000; Oakley et al. 2001; Sallnäs 2001; Hubbold, 2002; Jordan et al., 2002; Sallnäs and Zhai, 2003).

One study showed that subjects not only performed tasks significantly faster but also more precisely when manipulating objects together in a haptic compared to a nonhaptic collaborative virtual environment (Sallnäs et al., 2000; Sallnäs 2001). In this experiment tasks required that subjects lifted cubes by pushing from each side of the object in order to build different composed objects. It was found that people took significantly longer time to perform the five tasks in the visual only condition without haptic feedback than in the condition with haptic feedback (Sallnäs et al., 2000). It was also found that subjects made significantly more mistakes in performing the two tasks that required that subjects lifted cubes in the visual only than in the visual/haptic condition (Sallnäs 2001). Results also showed that when haptic force feedback was provided subjects’ perceived virtual presence was significantly improved but not their social presence (Sallnäs et al., 2000) measured by questionnaires. An explanation for this could be that haptic feedback improved the feeling of realism and control and interactivity, very much in accordance with Katz (1989) who argued that the touch is the primary sense for proof of realness, but that the audio contact over the phone channel is more important for social presence than haptic feedback is. It has been repeatedly shown that providing audio communication is the most important for increasing social behaviour but that for example adding video does not make as conclusive improvements. It was also shown that people reported that they performed the tasks and the collaboration significantly better when getting haptic feedback from the objects, the context and the other person’s movements.

Intuitively haptics may play a critical role when people pass objects between each other. The giver has to sense that the recipient has firmly grasped the object before releasing it. The recipient has to feel that the giver is releasing it before taking it towards oneself. It is difficult to imagine that such a task could be accomplished without haptic feedback. A study was performed in order to investigate how haptic force feedback affects people’s performance when handing over objects in a collaborative virtual environment (Sallnäs and Zhai, 2003). In an experiment, subjects passed a series of cubic objects to each other and tapped them at target areas. Their performance with and without haptic force feedback was evaluated. The subjects could not communicate verbally with each other during this experiment. Furthermore, the study was placed in the framework of Fitts’ law (Fitts, 1954) and it was hypothesized that object hand off constitutes a collaboratively performed Fitts’ law task, with target distance to target size ratio as a fundamental performance determinant. Results showed that task completion time indeed linearly increased with Fitts’ index of difficulty, both with and without force feedback. It was a little surprising that the time required for handing over objects did not differ significantly between the haptic and nonhaptic condition even though there was a large difference in favour of the haptic condition. However, the error rate was significantly lower with haptic feedback than without. Furthermore, it was found that people perceived virtual and social presence were significantly higher when collaborating in the visual/haptic condition. It was also found that haptic feedback significantly increased perceived performance when people performed a Fitts’ law tapping task collaboratively.

References

APPELLE, S. 1991. Haptic perception of form: Activity and stimulus attributes. In Heller, M. & Schiff, W. (Eds), *The psychology of touch.* New Jersey: Lawrence Erlbaum Associates, Inc. pp. 169-188.

BASDOGAN, C., HO, C., SLATER, M., AND SRINIVASAN, M. A. 1998. The Role of Haptic Communication in Shared Virtual Environments. *Proc. of the PHANTOM™ User Group.*

BIGGS, S. J., AND SRINIVASAN, M. A. 2002. Haptic interfaces. In K. M. Stanney (Ed.), Handbook of virtual environment technology. (pp. 93-116). Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

BURDEA, G. C. 1996. *Force and Touch Feedback for Virtual Reality*, John Wiley & Sons, Inc.

FITTS, P. M. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, **47**(6), 381-391.

GIBSON, J. J. 1979. The ecological approach to visual perception. Boston, MA: Houghton Miffin.

HELLER, M., AND SCHIFF, W. 1991. The psychology of touch. New Jersey: Lawrence Erlbaum Ass., Inc.

HUBBOLD, R. J. 2002. Collaborative stretcher carrying: A case study. The 8th Eurographics Workshop on Virtual Environments (pp. 7-12).

ISHII, M., NAKATA, M. AND SATO, M. 1994. Networked SPIDAR: A Networked Virtual Environment with Visual, Auditory, and Haptic Interactions. *Presence: Teleoperators and Virtual Environments.* 3(4), pp. 351-359.

JORDAN, J., MORTENSEN, J., OLIVEIRA, M., SLATER, M., AND SRINIVASAN, M. 2002. Collaboration in a mediated haptic environment. The 5th Annual Int. Workshop on Presence. Porto, Portugal.

KATZ, D. 1989. The world of touch. (L. E. Krueger, Trans.). Hillsdale, NJ: Lawrence Erlbaum Associates.

KATZ, R. AND TUSHMAN, M. 1978. Communication patterns, project performance, and task characteristics: An empirical evaluation in an R&D setting. *Organiza-tional Behavior and Human Performance*, 23 139-162.

LEDERMAN, S.J. AND KLATZKY, R.L. 1987. Hand movements: A window into haptic object recognition. *Cognitive Psychology*, **19**, 342-368.

LOOMIS, J. M. AND LEDERMAN, S. J. 1986. Tactual perception. In K.R. Boff, L. Kaufman, & J. P. Thomas (Eds.), Handbook of perception and human performance: Cognitive processes and performance (Vol.2). New York:Wiley.

MEHRABIAN, A. 1972. Methods and design: Some referents and measures of nonverbal behavior. Journal of Behav. Res. Meth. and Instrum., 1(6), 203-207.

OAKLEY, I., BREWSTER, S., AND GRAY, P. 2001. Can You Feel the Force? An Investigation of Haptic Collaboration in Shared Editors. In *Proceedings of Eurohaptics 2001*, Birmingham, UK, July 2001, C. Baber, M. Faint, S. Wall & A. M. Wing, Eds. University of Birmingham, Birmingham, UK, 54-59.

RUDDLE, R. A., SAVAGE, J.C.D. and JONES, D.M. (2002). Symmetric and asymmetric action integration during cooperative object manipulation in virtual environments. *ACM TOCHI.*

SALLNÄS, E-L., RASSMUS-GRÖHN, K., AND SJÖSTRÖM, C. 2000. Supporting presence in collaborative environments by haptic force feedback. *ACM TOCHI*, 7(4), 461-476.

SALLNÄS, E-L. 2001. Improved precision in mediated collaborative manipulation of objects by haptic force feedback. In G. Goos, J. Hartmanis and J. van Leeuwen (Series Eds.) and S. Brewster and R. Murray-Smith (Vol. Eds.), *LNCS: Vol. 2058. Haptic HCI* (pp. 69-75). Heidelberg, Germany: Springer.

SALLNÄS, E-L., AND ZHAI, S. 2003. Collaboration meets Fitts' law: Passing virtual objects with and without haptic force feedback. In M. Rauterberg, M. Menozzi & J. Wesson (Eds.), *Proceedings of INTERACT´03* (pp. 97-104). Amsterdam: IOS Press.

SHORT, J. WILLIAMS, E., AND CHRISTIE, B. (1976). *The social psychology of telecommunications.* London: Wiley.

Volume Haptics Technologies and Applications

Karljohan Lundin Palmerius*
Visual Information Technology and Applications
Linköping University, Sweden

1 Introduction

The human vision is often considered to be our most important channel of information. It is fast, intuitive and provides a high information bandwidth. Visualization technology and computer graphics, however, pushes the boundaries towards information overload. In these cases the introduction of haptics can be of great assistance, providing realistic interaction, natural constraints and improved control. It can also be used in a range of applications to intuitively reinforce visual information and provide guidance for increased speed and precision.

Applications and technologies for haptic interaction and feedback in the litterature typically based on the interaction with surfaces, such that we encounter in reality. These aim at providing a realistic copy of real stimuli. The power of computer generated haptic feedback, however, is not limited to mere surfaces. This presentation reviews applications and implementations that are empowered by the use of non-surface data or feedback metaphors.

2 Virtual Prototyping

Volumetric data can be used to describe real object for surface simulation. Since a volume is structured, algorithms applied on such data can be made much faster than on unstructured data, such as polygonal meshes. This effect has been applied in algorithms for virtual prototyping, for example in the *Voxmap Point-shell* method by McNeely in [McNeely 1993], where the static world is converted into a volumetric mesh and the haptic probe is converted into a point cloud. For each single point in the cloud the proximity and interaction with the imlicit surfaces in the volume can be effectively calculated and the combined effects can be used to simulated both translational and rotational feedback. This approach can also be made stable for most situations [Wan and McNeely 2003].

3 Scientific Visualization

The immense amount of information in scientific, volumetric data, such as the medical Computer Tomography (CT) and the simulated Computational Fluid Dynamics (CFD), can be overwhelming and overly complex in a purly visual interface. The addition of haptic cues and guidance has great potential to increase the capabilities of such interfaces. A formative study presented in [Lundin et al. 2006] identifies two primary uses of haptic feedback in volume visualization: information, both reinforcing visual impressions and providing complementary cues, and guidance, for finding and following features as well as mental guidance. To achieve this goal there exists a range of haptic representations of data.

A straightforward approach to provide *guidance* in volumetric data is to render a pushing or pulling force towards an area of interest or in the direction of the gradient vector, e.g. as presented in [Wall et al. 2002; Olofsson et al. 2004; Bartz and Gürvit 2000; Lawrence et al. 2004]. Several researchers have also described the use of the force metaphor to convey *information* about scalar data to a user. This approach was first introduced by Iwata et al. in [Iwata and Noma 1993] where a force in the direction of the gradient vector is used to represent the orientation of scalar data in combination with viscosity to represent the magnitude of the local scalars. A similar technique has been presented also by other research groups, e.g. in [Avila and Sobierajski 1996; Mor et al. 1996; Hashimoto and Iwata 1997]. From vector data a method to convey information is to use the vector as force feedback[Iwata and Noma 1993], although more advanced approaches have been proposed[Pao and Lawrence 1998; Donald and Henle 2000; Lawrence et al. 2004].

Another common way to provide feedback from scalar volume data is to extract an explicit local or global surface (e.g. [Körner et al. 1999]) from which classical surface haptics can be calculated, or to render the haptic feedback directly from the implicit representation of surfaces, e.g. as presented in [Salisbury and Tarr 1997; Thompson II et al. 1997]. A similar approach can be used to generate a shape representation from vector data[Lawrence et al. 2004]. It should be noted that by defining distinct shapes, every piece of data not part of that subset is unrepresented in the haptic rendering. Furthermore, haptic occlusion of potentially important areas is introduced by impenetrable shapes in the volume.

The yielding shapes as a representation of volumetric data was first introduced in [Lundin et al. 2002]. To avoid occlusion by impenetrable shapes these shapes are configured to yield if subjected to a force exceeding their appointed strengths. Thus, the strength is a property that can also be used to represent information in the data, for example how distinct a feature in the data is or how certain a shape estimation is. This approach has been used in volume visualization both to provide guidance (e.g. [Vidholm et al. 2004]) and information (e.g. [Lundin et al. 2005b]). A powerful implementation based on haptic primitives was presented in [Lundin et al. 2005a].

4 Tissue Simulation

While large body of research on surgery simulation concentrates on surface simulation for realistic deformations and speed, tissues are really much more than just surfaces. Volumetric representations of tissue data and accompanying algorithms are necessary to get realistic feedback in cases where the sense of touch is of importance. Examples can be found in simulation of both soft tissues and hard tissues, such as bone.

There exists two main approaches for bone drilling, the first one being similar to the Voxmap Point-shell method described above. This method was presented by Petersik et al. in [Petersik et al. 2002]. The volume mesh is here used to describe bone density instead of proximity to static surfaces. The other approach was introduced by Agus et al. and is described e.g. in [Agus et al. 2003]. Here the intersection between a spherical drill and the voxels in the volume describing the tissues is calculated. The sum of the values in the intersection is a measure of the drill's penetration of tissues and is used to estimate the feedback.

A typical example of soft tissue interaction where touch is of utmost importance is needle insertion and punctuation. In [Zhang et al. 2008], Zhang et al. use the haptic primitives described above to simulate the haptic feedback in a spinal anaesthesia needle insertion simulator. The haptics cues that indicate tissue penetration can be felt and the correct procedure trained.

*e-mail: karlu@itn.liu.se

References

AGUS, M., GIACHETTI, A., GOBBETTI, E., ZANETTI, G., AND ZORCOLO, A. 2003. Real-time haptic and visual simulation of bone dissection. *Precense: Teleoperators and Virtual Environments 12*, 1, 110–122.

AVILA, R. S., AND SOBIERAJSKI, L. M. 1996. A haptic interaction method for volume visualization. In *Proceedings of IEEE Visualization*, 197–204.

BARTZ, D., AND GÜRVIT, Ö. 2000. Haptic navigation in volumetric datasets. In *Proceedings of PHANToM User Research Symposium*.

DONALD, B. R., AND HENLE, F. 2000. Using haptics vector fields for animation motion control. In *Proceedings of IEEE International Conference on Robotics and Automation*.

HASHIMOTO, W., AND IWATA, H. 1997. A versatile software platform for visual/haptic environment. In *Proceedings of ICAT*, 106–114.

IWATA, H., AND NOMA, H. 1993. Volume haptization. In *Proceedings of IEEE 1993 Symposium on Research Frontiers in Virtual Reality*, 16–23.

KÖRNER, O., SCHILL, M., WAGNER, C., BENDER, H.-J., AND MÄNNER, R. 1999. Haptic volume rendering with an intermediate local representation. In *Proceedings of the 1st International Workshop on the Haptic Devices in Medical Applications*, 79–84.

LAWRENCE, D. A., PAO, L. Y., LEE, C. D., AND NOVOSELOV, R. Y. 2004. Synergistic visual/haptic rendering modes for scientific visualization. *IEEE Computer Graphics and Applications 24*, 6, 22–30.

LUNDIN, K., YNNERMAN, A., AND GUDMUNDSSON, B. 2002. Proxy-based haptic feedback from volumetric density data. In *Proceedings of the Eurohaptic Conference*, University of Edinburgh, United Kingdom, 104–109.

LUNDIN, K., GUDMUNDSSON, B., AND YNNERMAN, A. 2005. General proxy-based haptics for volume visualization. In *Proceedings of the IEEE World Haptics Conference*, IEEE, 557–560.

LUNDIN, K., SILLEN, M., COOPER, M., AND YNNERMAN, A. 2005. Haptic visualization of computational fluid dynamics data using reactive forces. In *Proceedings of the Conference on Visualization and Data Analysis, part of IS&T/SPIE Symposium on Electronic Imaging 2005*.

LUNDIN, K., COOPER, M., PERSSON, A., EVESTEDT, D., AND YNNERMAN, A. 2006. Enabling design and interactive selection of haptic modes. *Virtual Reality 11*, 1 (March), 1–13. DOI: 10.1007/s10055-006-0033-7.

MCNEELY, W. A. 1993. Robotic graphics: A new approach to force feedback for virtual reality. In *Proceedings of IEEE Virtual Reality Annual International Symposium*.

MOR, A., GIBSON, S., AND SAMOSKY, J. 1996. Interacting with 3-dimensional medical data: Haptic feedback for surgical simulation. In *Proceedings of Phantom User Group Workshop*.

OLOFSSON, I., LUNDIN, K., COOPER, M., KJÄLL, P., AND YNNERMAN, A. 2004. A haptic interface for dose planning in stereo-tactic radio-surgery. In *Proceedings of the 8th International Conference on Information Visualization '04*, IEEE, 200–205.

PAO, L., AND LAWRENCE, D. 1998. Synergistic visual/haptic computer interfaces. In *Proceedings of Japan/USA/Vietnam Workshop on Research and Education in Systems, Computation, and Control Engineering*.

PETERSIK, A., PFLESSER, B., TIEDE, U., HÖHNE, K. H., AND LEUWER, R. 2002. Realistic haptic volume interaction for petrous bone surgery simulation. In *CARS*.

SALISBURY, K., AND TARR, C. 1997. Haptic rendering of surfaces defined by implicit functions. In *Proceeding of the ASME 6th Annual Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*.

THOMPSON II, T. V., JOHNSON, D. E., AND COHEN, E. 1997. Direct haptic rendering of sculptured models. In *Proceedings of Symposium on Interactive 3D Graphics, Providence, RI.*

VIDHOLM, E., TIZON, X., NYSTRÖM, I., AND BENGTSSON, E. 2004. Haptic guided seeding of MRA images for semi-automatic segmentation. In *Proceedings of IEEE International Symposium on Biomedical Imaging*.

WALL, S. A., PAYNTER, K., SHILLITO, A. M., WRIGHT, M., AND SCALI, S. 2002. The effect of haptic feedback and stereo graphics in a 3D target acquisition task. In *Proceedings of Eurohaptics*, University of Edinburgh, United Kingdom.

WAN, M., AND MCNEELY, W. A. 2003. Quasi-static approxmation for 6 degrees-of-freedom haptic rendering. In *Proceedings of IEEE Visualization, pp. 257–262.*

ZHANG, D., ALBERT, D., HOCKEMEYER, C., BREEN, D., KULCSÁR, Z., SHORTEN, G., ABOULAFIA, A., AND LÖVQUIST, E. 2008. Developing competence assessment procedure for spinal anaesthesia. In *IEEE International Symposium on Computer-Based Medical Systems*, 397–402.

Learning Molecular Interaction Concepts through Haptic Protein Visualization

Petter Bivall Persson*
Department of Science and Technology, Linköping University, Sweden

Abstract

The use of haptics is growing in the area of science education. Haptics appears to convey information to students in a manner that influences their learning and ways of thinking. This document outlines examples of how haptics has been employed in science education contexts and gives a more detailed description of an education oriented evaluation of a haptic protein-ligand docking system.

In molecular life science, students need to grasp several complex concepts to understand molecular interactions. Research on how haptics influences students' learning show strong positive affective responses and, in the protein-ligand docking case, that reasoning with respect to molecular processes is altered. However, since many implications of using haptics in education are still unknown, more research is needed.

CR Categories: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities; H.5.2 [Information Interfaces and Presentation]: User Interfaces—Haptic I/O; K3.1 [Computer Uses in Education]: Computer-assisted instruction— [H.3.4]: Systems and Software—Performance evaluation (efficiency and effectiveness)

Keywords: haptics, molecular visualization, science education research, protein-ligand docking

1 Introduction

Molecular bioscience employs the widespread use of visual representations that range from sketches to advanced computer graphics (CG). There is a need to convey abstract knowledge and to conceptualize complex three-dimensional relationships within and between molecules. However, recent research suggests that students have great difficulty understanding these concepts, despite the use of visualizations aimed at making concepts more comprehensible [Wu et al. 2001].

To investigate the challenges and possibilities of utilizing haptics in life science, we have developed a haptic protein-ligand docking application called Chemical Force Feedback (CFF), which allows the user to manipulate the ligand (a small molecule) and feel its interactions with the protein in the docking process.

The CFF system was developed and evaluated in a collaborative and inter-disciplinary research effort, involving the author, Anders Ynnerman (Prof.) and Matthew D. Cooper (Senior Lecturer) at the Department of Science and Technology, Lena Tibell (Associate Prof.) at the Department of Clinical and Experimental Medicine, Bengt-Harald Jonsson (Prof.) and Gunnar Höst (PhD) at the Department of Physics, Chemistry and Biology, all of the above working at Linköping University, and Shaaron Ainsworth (Associate Prof.) in the School of Psychology at the University of Nottingham.

The next sections include a general description of haptic protein-ligand docking and briefly overviewing examples showing how haptics has been employed within science education. This is followed by a more detailed description of the education oriented evaluation of the CFF system and its results.

2 Haptic Protein-Ligand Docking

In 1967 the GROPE project [Brooks Jr. et al. 1990; Ouh-young et al. 1988] started their research on using haptic feedback to convey molecular interaction forces. The project evolved to include protein-ligand docking in its third version, GROPE III. Since then, several haptic protein-ligand docking systems have been developed and presented in research publications, for example [Sankaranarayanan et al. 2003; Weghorst 2003; Lee and Lyons 2004; Lai-Yuen and Lee 2005; Bivall Persson et al. 2007; Wollacott and Merz Jr. 2007; Daunay et al. 2007; Subasi and Basdogan 2008]. However, most studies do not discuss the explicit use of haptics in educational settings and extensive evaluations are very rare.

In protein-ligand docking the user, or an automated docking simulation, searches for the configuration of the ligand that place it in a global energy minimum. This is a difficult minimization problem with $6 + n$ degrees of freedom, 6 for position and orientation and n for the number or rotational bonds in the ligand. Basically, the energy can be calculated by pairwise atom-atom interactions, but the high number of atoms in most proteins makes that approach unfeasible in haptic systems as they require high update rates (approximately 1kHz). The most common approach used to achieve the required update rate is to use static potential grid maps that represent the potential field of the protein as volumetric data. Forces acting on the atoms in the ligand can be derived from the gradient of the potential field, making computational speed constrained by the number of atoms in the ligand alone. There have been attempts to include dynamic models of proteins, most of which struggle with computational speed that limits the number of atoms that can be used in the molecules. However, as models for molecular dynamics and processor speeds continue to improve, the docking systems will move toward full real-time simulations.

3 Haptics in Science Education Contexts

Novel technology can provide possibilities for new approaches in teaching and learning. Biology and chemistry education has embraced visualization technology as it is expected to aid the understanding of molecular structure and interaction. However, there is still a lot of work to be done to investigate which types of representations best convey information about molecular life science, and in determining the conditions for beneficial use of haptic and visual applications in education. Research regarding the use of haptics in educational settings seems to conclude that force feedback can ease the understanding of a variety of complex processes, especially when dealing with concepts that include elements of forces we handle in our everyday life, or when the mapping between the studied phenomenon and force is intuitive [Křenek et al. 1999]. Also, Druyan [Druyan 1997] has shown that and the ability to use kinaesthetics may help in grasping concepts concerning physical phenomena. A review of the research done to investigate the efficacy of haptics in education is presented in [Minogue and Jones 2006] and in the remainder of this section there will be a short presentation of examples where haptics has been used in education.

Reiner [Reiner 1999] investigated students' understanding and development of force concepts during interaction with a system that provided 2D visual and tactile feedback, allowing learners to feel different force fields. Results of the study suggested that the hap-

*e-mail: petbi@itn.liu.se

tic feedback induced learners to construct representations of forces that were closely related to the representations of formal physics.

Jones et al. [Jones et al. 2003; Jones et al. 2006] investigated how the use of haptics influences students' interpretation of viruses at the nanoscale. Their research showed that haptics influences learners' construction of concepts and their level of engagement. Students exposed to haptics used more affective terms, expressing that they were more interested in the subject and that they felt like they could participate more fully in the experience. These affective responses can be very important in learning contexts.

Another example where haptics appears to aid the understanding of a complex concept is presented in [Harvey and Gingold 2000], where an electron density function (an $R^3 \rightarrow R$ function) is used as basis for the generated force feedback. It is stated that the electron density function is hard for students to grasp, and that images do not suffice, often leading to misconceptions because a representation of the function requires four dimensions. Haptics is found to ease understanding and avoid oversimplifications by translating the fourth dimension to force.

In [Sankaranarayanan et al. 2003; Weghorst 2003] an application of augmented reality is presented, mixing a physical molecular model with CG. The system described consists of three major parts: augmented reality (AR), a voice command interface and force feedback. Haptics appears to provide a natural and intuitive method for representing the interaction between molecules, voice commands enables more focus on the model and less on the computer interface, and the combination of a physical model and CG through augmented reality is an attempt to take advantage of the best of two worlds, the physical and the virtual.

Tests with a few cases are presented, for example a physical model of the HIV protease augmented with CG generated inhibitors, and a physical protein model combined with visualization of its potential field. A lesson plan was created to evaluate the usefulness of the model in teaching, and an assessment was performed with a biology class. The assessment shows the model to be engaging and instructive and the AR model is received by both scientists and students as being very intuitive. However, a more developed lesson plan is required for it to be more effective.

4 A Haptic Docking System Evaluated In situ

In [Bivall Persson et al. 2007] we presented an evaluation of the CFF haptic docking system. We used a combined quantitative and qualitative assessment in an *in situ* learning situation, placing focus on the quality of learning and understanding of the molecular interactions. The aim was to evaluate the CFF system's features, its interface and usability, but specifically to explore the impact of haptics on students' performance and learning, investigating what the haptic modality adds to a visual 3D protein structure representation in this context.

The study was performed with life science and engineering students enrolled in a course at Linköping University, called *Biomolecular interactions*. The course deals with the interactions between proteins and ligands, factors determining structure recognition and the dynamics of molecular exchange. To understand these processes the students have to grasp several concepts of varying complexity, for example molecular structure, affinity, specificity, energy levels, binding, molecular dynamics and transition state. Students carried out computer based lab exercises using the CFF haptic environment. The labs were a compulsory part of the course, although participation in the research was voluntary.

4.1 Test Design

A partial cross-over test design was employed, partly because of the limited number of students (13 females and 10 males). The subjects were divided into two groups, *0/H* and *H/0*, according to gender and score on an initial domain knowledge test, aiming at an even distribution with respect to gender and achievement levels. The group names include a condition coding where *0* denotes no haptics and *H* indicate active haptic feedback.

In two lab tasks the students were required to attempt to find the best docking to the enzyme *human carbonic anhydrase II*, using ligands that produced different force binding strengths to the enzyme. Both groups performed the labs with the CFF system but the *H/0* group performed the first task (*Task IS*) with force feedback enabled, whereas it was disabled for the *0/H* group. For the second task (*Task TS*) the condition between the groups was reversed. Both groups were probed according to the following time-line:

1. Background survey
2. Pre-test Task IS
3. Lab exercise IS
4. Post-test Task TS
5. Items 2-4 repeated for Task TS
6. Experience survey
7. Interview with a subset of students

Pre- and post-tests were designed to enable a measure of the cognitive gain from the use of the haptic representation. Additionally, written answers to the lab tasks were available for qualitative analysis, and a subset of the students were chosen (based on achievement levels) for semi-structured clinical interviews [Ginsburg 1997; Kvale 1996]. Interviews were centred on cognitive understanding, affective factors and opinions, as well as on meaning making and the use of the haptic representation while solving a docking problem.

To ensure a reliable assessment of responses to lab tasks as well as to pre- and post-tests, two teachers/scientists individually marked the students' answers. The equally assessed responses covered more than 95% of the total, indicating a strong reliability consistency. Reasoning in the task responses and interviews was analyzed using analytical induction, a process involving repeated readings of the responses and interview transcripts, focusing on key terms of the subject.

The students' docking performance was scored by comparison with automated docking results (as calculated by AutoDock [Goodsell and Olson 1990; Morris et al. 1996; Morris et al. 1998]) using a Root-Mean-Square error (RMS, expressed in Ångström), a common technique used to compare alignment of molecules.

4.2 Results of the Study

Survey data showed that the affective responses to the docking experience was clearly positive and that students found the haptic feedback to be helpful. Interviewed students also expressed that the haptic system aided their understanding by allowing them to connect different parts of their knowledge in a more coherent way.

By comparing scores on the students' pre- and post-tests it was found that the students learned from their experience with the CFF system. However, the scores did not reveal a significant difference between the conditions (*H/0, 0/H*). Reasoning, on the other hand, was influenced by the condition as students using haptics showed more reasoning regarding forces and dynamics.

5 Conclusion

Based on the low number of cases where haptics has been applied in science education, it can be argued that there is a lack of empirical evaluations performed in educational settings, a conclusion also reached in [Minogue and Jones 2006].

The examples presented in section 3 provide an indication for how haptics can be beneficial in learning situations. Our study also shows that the implications go somewhat beyond mere speed gains and positive affective responses and induce new ways of reasoning. Nevertheless, the use of haptics in education is largely an uncharted territory and further research is required to understand how haptics should be applied to support learning and teaching.

References

BAYAZIT, O. B., SONG, G., AND AMATO, N. M. 2000. Ligand binding with obprm and haptic user input: Enhancing automatic motion planning with virtual touch. Tech. Rep. TR00-025, Department of Computer Science, Texas A&M University, Texas, USA, October.

BIVALL PERSSON, P., COOPER, M. D., TIBELL, L. A., AINSWORTH, S., YNNERMAN, A., AND JONSSON, B.-H. 2007. Designing and evaluating a haptic system for biomolecular education. In *Proceedings of IEEE Virtual Reality 2007*, IEEE, 171–178.

BROOKS JR., F. P., OUH-YOUNG, M., BATTER, J. J., AND KILPATRICK, P. J. 1990. Project GROPE - haptic displays for scientific visualization. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 177–185.

DAUNAY, B., MICAELLI, A., AND RGNIER, S. 2007. 6 DOF haptic feedback for molecular docking using wave variables. In *2007 IEEE International Conference on Robotics and Automation*, 840–845.

DRUYAN, S. 1997. Effect of the kinesthetic conflict on promoting scientific reasoning. *Journal of Research in Science Teaching 34*, 10 (December), 1083–1099.

GINSBURG, H. P. 1997. *Entering the Child's Mind: The Clinical Interview In Psychological Research and Practice*. Cambridge University Press, Cambridge.

GOODSELL, D. S., AND OLSON, A. J. 1990. Automated docking of substrates to proteins by simulated annealing. *Proteins: Structure, Function, and Genetics 8*, 3, 195–202.

HARVEY, E., AND GINGOLD, C. 2000. Haptic representation of the atom. In *Proceedings of the International Conference on Information Visualisation 2000*, 232–235.

JONES, M. G., ANDRE, T., SUPERFINE, R., AND TAYLOR, R. 2003. Learning at the nanoscale: The impact of students' use of remote microscopy on concepts of viruses, scale, and microscopy. *Journal of Research in Science Teaching 40*, 3 (March), 303–322. Haptic feedback, physical paper and clay models.

JONES, M. G., MINOGUE, J., TRETTER, T. R., NEGISHI, A., AND TAYLOR, R. 2006. Haptic augmentation of science instruction: Does touch matter? *Science Education 90*, 1 (January), 111–123.

KVALE, S. 1996. *InterViews: An Introduction to Qualitative Research Interviewing*. Sage Publications, Inc, Thousand Oaks.

KŘENEK, A., ČERNOHORSK, M., AND KABELČ, Z. 1999. Haptic visualization of molecular model. In *WSCG'99 Conference Presentation*, V. Skala, Ed., vol. 1, 133–139.

LAI-YUEN, S. K., AND LEE, Y.-S. 2005. Computer-aided molecular design (camd) with force-torque feedback. In *Ninth International Conference on Computer Aided Design and Computer Graphics*, IEEE Computer Society, Los Alamitos, CA, USA, 199–204.

LEE, Y.-G., AND LYONS, K. W. 2004. Smoothing haptic interaction using molecular force calculations. *Computer-Aided Design 36*, 1 (January), 75–90.

MINOGUE, J., AND JONES, M. G. 2006. Haptics in education: Exploring an untapped sensory modality. *Review of Educational Research 76*, 3, 317–348.

MORRIS, G. M., GOODSELL, D. S., HUEY, R., AND OLSON, A. J. 1996. Distributed automated docking of flexible ligands to proteins: Parallel applications of autodock 2.4. *Journal of Computer-Aided Molecular Design 10*, 10, 293–304.

MORRIS, G. M., GOODSELL, D. S., HALLIDAY, R. S., HUEY, R., HART, W. E., BELEW, R. K., AND OLSON, A. J. 1998. Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function. *Journal of Computational Chemistry 19*, 14, 1639–1662.

NAGATA, H., MIZUSHIMA, H., AND TANAKA, H. 2002. Concept and prototype of protein-ligand docking simulator with force feedback technology. *Bioinformatics 18*, 1, 140–146.

OUH-YOUNG, M., PIQUE, M., HUGHES, J., SRINIVASAN, N., AND BROOKS JR, F. P. 1988. Using a manipulator for force display in molecular docking. In *Proceedings of IEEE Robotics and Automation Conference 3*, 1824–1829.

REINER, M. 1999. Conceptual construction of fields through tactile interface. *Interactive Learning Environments 7*, 1 (April), 31–55.

SANKARANARAYANAN, G., WEGHORST, S., SANNER, M., GILLET, A., AND OLSON, A. 2003. Role of haptics in teaching structural molecular biology. In *Proceedings of the 11th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (IEEE HAPTICS'03)*, 363–366.

SUBASI, E., AND BASDOGAN, C. 2008. A new haptic interaction and visualization approach for rigid molecular docking in virtual environments. *Presence: Teleoperators and Virtual Environments 17*, 1, 73–90.

WEGHORST, S. 2003. Augmented tangible molecular models. In *IWSM'03-'04: The Fourth International Workshop on Spatial Media and the Sixth International Conference on Humans and Computers*.

WOLLACOTT, A. M., AND MERZ JR., K. M. 2007. Haptic applications for molecular structure manipulation. *Journal of Molecular Graphics and Modelling 25*, 6 (March), 801–805.

WU, H.-K., KRAJCIK, J. S., AND SOLOWAY, E. 2001. Promoting understanding of chemical representations: Students' use of a visualization tool in the classroom. *Journal of Research in Science Teaching 38*, 7 (September), 821–842.

Physically Realistic Soft-Tissue Deformation in Real-Time

Umut Koçak*
[1]Norrköping Visualisation and Interaction Studio, Linköping University
SWEDEN

Abstract

The use of haptics in virtual applications has become widespread in the last decade in several areas, especially in medical applications. After the integration of haptics, virtual medical simulations have been more popular than plastic tissue models or cadavers. Virtual environments are better for not only objective performance assesment but also creating desired scenarios in medical hands-on training. Besides various surgery scenarios can be created as well as surgery rehearsals by using patient specific data in virtual environments. However introduction of haptics into virtual simulations has also triggered new challenges in the area because of the high refresh rates needed for the haptics.

The sufficient refresh rate for the haptic force is 1 kHz such that discontinuity is not perceived by human sense. Besides the deformable objects need to be modeled in high complexity to provide realistic behaviour especially in surgery simulations. Therefore the simulation of deformable soft tissues with physically realistic force feedback in real time is still one of the big challenges in the haptics area. The major challenge is the computational burden to simulate realistic deformations which cannot be easily handled by using even today's fastest computers. A compromise between the realism and speed has to be achieved in the simulation of soft tissue deformation. There are two general methods followed in the literature: mass-spring and finite element models. Even though mass-spring models are easy to implement and sufficient refresh rates can be reached; the physical properties of materials cannot be taken into account and the model may respond un-realistically to larger deformations in these methods. Although the latest studies reveal that it is possible to simulate physically realistic deformations by using Finite Element Methods (FEM), the real time requirement is not easy to satisfy because of the complexity of this approach.

In spite of the computational load of FEM, it is still the most popular method to model physically realistic deformations for soft tissue. While it is easier to solve linear FEM equations in real-time, it is observed that soft tissues have non-linear behaviour, such that the linear deformation becomes un-realistic if the size of the deformation exceeds a threshold (typically 10% of the original tissue size). In addition to non-linearity, soft tissues may also have various complex behaviours like visco-elasticity and anisotropicity increasing the computational burden. If tissue cutting is also to be simulated, the model should be updated both physically and virtually, which makes the real time simulations even harder.

To reach a compromise between the necessity of high haptic refresh rates and the computational burden of the FEM, several optimization techniques such as condensation [Bro-Nielsen and Cotin 1996], pre-computation [Cotin et al. 1999; Sedef et al. 2006; Sela et al. 2007], level of detail [Debunne et al. 2001], and exploitation of the sparse matrix structure have been introduced in simulations. Current simulators, however, still have to either sacrifice one property of real tissues such as non-linearity, anisotropicity or visco-elasticity, or apply force interpolation or extrapolation techniques to reach a sufficient haptic refresh rate (1 kHz). Therefore solution of large FEM systems in real-time to be used in soft-tissue deformations is still a challenge for which new solutions are being sought.

Keywords: Soft tissue deformation, surgery simulation, FEM

References

BRO-NIELSEN, M., AND COTIN, S. 1996. Real-time volumetric deformable models for surgery simulation using finite elemets and condensation. In *Computer Graphics Forum*, 57–66.

COTIN, S., DELINGETTE, H., AND AYACHE, N. 1999. Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions on Visualization and Computer Graphics*, 62–73.

DEBUNNE, G., DESBRUN, M., CANI, M.-P., AND BARR, A. H. 2001. Dynamic real-time deformations using space and time adaptive sampling. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 31–36.

SEDEF, M., SAMUR, E., AND BASDOGAN, C. 2006. Real-time finite-element simulation of linear viscoelastic tissue behavior based on experimental data. *IEEE Computer Graphics and Applications*, 58–68.

SELA, G., SUBAG, J., LINDBLAD, A., ALBOCHER, D., SCHEIN, S., AND ELBER, G. 2007. Real-time haptic incision simulation using fem-based discontinuous free-form deformation. *Computer Aided Design*, 685–693.

*e-mail: umut.kocak@itn.liu.se

Simulator for Operative Extraction of Wisdom Teeth

Jonas Forsslund*
Royal Institute of Technology

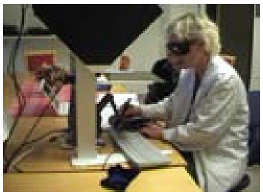


Figure 1: *Oral surgeon in cooperative evaluation*

Abstract

Hands-on practical experience in oral surgery is highly requested by dental students and faculty. For the purpose of training, a haptic enabled simulator for surgical extraction of wisdom teeth was designed. A prototype was implemented and evaluated with surgeons from Karolinska Institute as part of a user centered design approach.

Keywords: haptics, medical simulation, user centered design

1 Background

Dental education in Sweden covers the theoretical aspects of oral surgery, but students unfortunately receive no or negligible practical practice due to limited resources. However, hands-on practical experience in surgical procedures is highly requested by students. The purpose of the research is to develop, through an applied user-centered design approach, a simulator that allows students to practice and perfect surgical procedures for wisdom tooth extraction before operating on live patients.

2 Surgery simulators

Some of the difficulties associated to the creation of a haptic enabled surgery simulator are the real-time constraints. A successful simulator requires real-time interaction with virtual organs via a force feedback interface in combination with real time visualization of the applied deformations. Simulators work best with multimodal displays, especially audio combined with haptics [Roberts and Panels 2007].

Commercially successful medical simulators exists in the field of endoscopy training. It has been proven that simulator based training of the required visual-spatial and psycho-motor skills required for such procedures significantly improves the performance of students [Fellnder-Tsai and Wredmark 2004]. The improvement of students performance can be measured in time and in some procedures as level of patient discomfort. It has been shown that students perform faster and with less patient discomfort in the first live patient examinations after undergoing simulator based training [Ahlberg et al. 2005].

Bone and dental surgery simulators have not yet been as extensively evaluated, but there are numerous research projects related to applications in bone surgery simulation technology. The main tool of use with these kind of simulators is the rotating drill and the object of deformation is hard tissue. For example research at Stanford focuses on both mandibular surgery and temporal bone surgery [Morris et al. 2006].

3 User centered design

In this project, a user centered method based on the ISO 13407 was used. The ISO standard model comprises of four design activities that form one iteration, which is by definition [ISO 1999]:

1. Understand and specify the context of use
2. Specify the user and organisational requirements
3. Produce design solutions
4. Evaluate design against requirements

The method was applied by conducting a Contextual Inquiry comprising of interviews, surgery observations and hands-on drilling experiments, followed by implementation of a prototype simulator that was evaluated with surgeons. The process should be considered the first development iteration, where results from the evaluation is input to the following design iteration.

4 Simulator implementation

An immersive setup was used, see figure 1, where the user control the handle of a Sensable Phantom haptic device while looking at a monitor through a mirror, positioned so that the handle is co-located with the image of the virtual drill. Stereo shutter glasses provides the user with depth vision and the simulation ran on a Pentium 4, 3.4 GHz computer equipped with a Nvidia Quadro FX1400 (128mb) graphics card. The software was implemented in C++ based on SenseGraphics open source API's.

The contextual inquiry suggested that it was of great importance to visually constrain the work space since the mimicked environment context was a patient's mouth. The context of choice was a polygon mesh of a standard head, modified to suit the particular surgical task. In the model's mouth, a carvable jaw part was placed that contain a partially impacted wisdom tooth and proximity bone and teeth, as seen in Figure 2.

4.1 Haptic rendering

The three degree of freedom haptic device simulates a one-point interaction and therefore needs active hand movement to provide

*e-mail: jofo02@kth.se

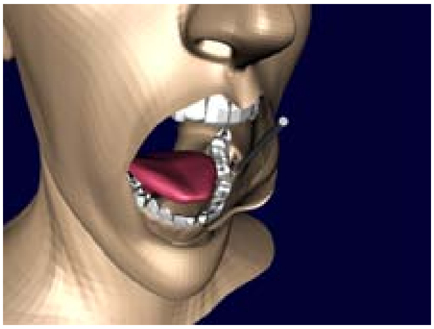


Figure 2: *Screenshot of the volume rendered jaw part placed in a polygon mesh model.*

exploration of a virtual surface. In our case the virtual surface is the human jaw, and the tool of exploration is the drill.

Virtual shapes are simulated by rendering forces to the haptic device in an update rate of normally 1000 hz. The basis for the haptic algorithm implemented in this project is a volume intersection model developed by Marco Agus, where the force magnitude and direction are calculated from the sampled volume intersection of the spherical drill and the volume model [Agus 2004].

The carvable jaw part was cropped from a 512x512x277 scalar field volume obtained from a Computed Tomography scan of a live patient's head. An attenuation threshold classification method was used to binary classify voxels as part of material or not. Material was considered as attenuation values above soft bone, lower values was considered air (although most was flesh etc). Since the classification was the basis for the force calculation, having only one threshold value creates artifacts perceived as an undulated rough surface. Given the stochastic nature of the x-ray attenuation, a one-sided Gauss filtering was applied which reduced the roughness [Forsslund 2008].

4.2 Graphical rendering

The same jaw part volume used for haptic rendering was rendered visually by a ray cast based volume rendering method provided by API. An opacity transfer function was used, but with no explicit segmentation. The model was updated in almost real time without further optimization. The base for the contextual polygon mesh model was obtained from a modeling tool called MakeHuman and modified in standard 3D modeling software, Blender.

5 Evaluation

The goal of the evaluation was to verify design decisions and get input for modifications in future iterations. Cooperative Evaluation is a method where the evaluator sits next to the user, who has been given a task to perform but relative freedom of operation. It is important to get the user to provide constant feedback in order to capture the user's goal and if the system's response is as expected [Monk et al. 1993].

A first Cooperative Evaluation of the prototype with four surgeons from Karolinska Institute was conducted in late 2007. The task was, after a preliminary free exploration of the simulator's capabilities, to perform the first mandibular bone carving procedure of the wisdom tooth extraction surgery. The study was performed on a protocol based on Appendix 1 of Monk et al's work [Monk et al. 1993].

6 Results

The Contextual Inquiry revealed that tacit knowledge that surgeons rely on, using vision, hearing and touch was essential in surgery. For vision, the very constrained environment of the mouth implied the design of the context mesh has to be taken into account to accurately reflect the actual surgical environment. The interviews suggested that simulated audio was not important, but later proved to have greater importance in the evaluation. Overall, the results of the Cooperative Evaluation showed that training by virtual jaw bone drilling was possible.

The source code for this simulator has been rewritten and released as free and open source software under the name "forssim".

References

AGUS, M. 2004. *Haptic and Visual Simulation of Bone Dissection.* PhD thesis, Dept. of Mechanical Engineering, University of Cagliari, Italy.

AHLBERG, G., HULTCRANTZ, R., JARAMILLO, E., LINDBLOM, A., AND ARVIDSSON, D. 2005. Virtual reality colonoscopy simulation: A compulsory practise for the future colonoscopist? *Endoscopy 37*, 12 (December), 1198–1204.

FELLNDER-TSAI, L., AND WREDMARK, T. 2004. Image-guided surgical simulation - a proven improvement. *Acta Orthop Scand 75*, 5, 515–515.

FORSSLUND, J. 2008. *Simulator för operativ extraktion av visdomständer.* Master's thesis, Kungliga Tekniska Hgskolan.

ISO. 1999. *Human-centred design processes for interactive systems (ISO 13407:1999).* International Organization for Standardization.

MONK, A., WRIGHT, P., HABER, J., AND DAVENPORT, L. 1993. *Improving you human-computer interface: a practical technique.* A volume in the BCS Practitioner Series, Prenice-Hall, ISBN 0-13-010034-X.

MORRIS, D., SEWELL, C., BARBAGLI, F., SALISBURY, K., BLEVINS, N. H., AND GIROD, S. 2006. Visuohaptic simulation of bone surgery for training and evaluation. *IEEE Transactions on Computer Graphics and Applications* (November), 48–57.

ROBERTS, J. C., AND PANELS, S. 2007. Where are we with haptic visualization? In *WorldHaptics 2007, Second Joint EuroHaptics Conference, 2007 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 316–323.

Papers

Fast and Tight Fitting Bounding Spheres

Thomas Larsson*
School of Innovation, Design and Engineering
Mälardalen University
Sweden

Abstract

Bounding spheres are utilized frequently in many computer graphics and visualization applications, and it is not unusual that the computation of the spheres has to be done during run-time at real-time rates. In this paper, an attractive algorithm for computing bounding spheres under such conditions is proposed. The method is based on selecting a set of k extremal points along $s = k/2$ input directions. In general, the method is able to compute better fitting spheres than Ritter's algorithm at roughly the same speed. Furthermore, the algorithm computes almost optimal spheres significantly faster than the best known smallest enclosing ball methods. Experimental evidence is provided which illustrates the qualities of the approach as compared to five other competing methods. Also, the experimental result gives insight into how the parameter s affects the tightness of fit and computation speed.

CR Categories: F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems—Geometrical problems and computations; I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling

Keywords: bounding sphere, enclosing ball, extremal points, computational geometry, computer graphics

1 Introduction

Bounding spheres, also called enclosing balls, are often used to accelerate computations in computer graphics, GIS, robotics, and computational geometry. For example, they are commonly utilized to speed up operations such as picking, ray tracing, view-frustum culling, collision detection, motion planning, and range queries. Efficient computation of bounding spheres is therefore of great importance.

In 3D, the minimum sphere enclosing a set of points is uniquely defined by 2, 3, or 4 points. These points are called the set of support since they are extremal points lying on the surface of the sphere. There can be more than 4 points on the surface, but at most 4 points are needed to uniquely define the smallest sphere. Thus, a brute force computation of the minimal sphere can be performed by considering all possible combinations of 2, 3, and 4 points. For each combination, a minimal enclosing sphere is computed. Then the sphere is checked to see if it also encloses all other points. If this is the case, and the computed sphere is also the smallest valid bounding sphere found so far, it is kept as the currently smallest bounding sphere. When all the combinations have been processed, it is guaranteed that the minimal bounding sphere has been found. Of course, such a brute force $O(n^5)$ method is prohibitively slow.

Interestingly, Megiddo has presented a deterministic linear time algorithm. However, an actual implementation of this algorithm is likely to be slow due a large hidden constant in the time complexity [Megiddo 1983]. The randomization method by Welzl, however, runs in expected $O(n)$ time, and several robust implementations have been presented [Welzl 1991; Gärtner 1999; Eberly 2007]. Unfortunately, implementations based on Welzl's algorithm [1991] are still considered too slow in many real-time applications. As it appears, simpler and faster algorithms are often preferred, despite the fact that non-optimal bounding spheres are computed [Ritter 1990; Wu 1992; Ericson 2005]. Ritter's algorithm, for example, is an extremely fast two-pass linear time algorithm, which has become very popular [Ritter 1990]. Also, approximation methods based on core-sets have been proposed [Bâdoiu and Clarkson 2003; Kumar et al. 2003; Bădoiu and Clarkson 2008]. These methods are able to compute $(1 + \epsilon)$-approximations of the smallest enclosing sphere, and they have been reported to be considerably faster than the best known exact solvers.

We propose a fast and simple algorithm called the *Extremal Points Optimal Sphere* (EPOS) to compute tight fitting bounding spheres in worst-case $O(sn)$ time. The input variable s, which determines the number of directions (normal vectors) considered by the algorithm internally, can be varied as a mean to balance the trade-off between execution time and sphere quality. Using a small constant value of, e.g. $s = 3$, yields a highly efficient $O(n)$ algorithm, which computes tight fitting spheres in many cases. In fact, in this case, the algorithm runs approximately as fast as Ritter's method, while still producing bounding spheres of higher quality. Furthermore, like the Bădoiu-Clarkson algorithm, the proposed method is always capable of computing close to optimal spheres, while being significantly faster.

2 The EPOS Algorithm

A bounding sphere, S, of a point set P with n points is described by a center point, **c**, and a radius, r. As stated earlier, there is always a unique bounding sphere that is minimal. To compute it, we need to locate 2, 3, or 4 supporting points. The algorithm attempts to locate these points quickly or at least a good approximation of them. The pseudocode is given in Figure 1. It starts out by selecting a suitable subset E of P with k extremal points such that $4 < k < n$ (Line 2). Then the minimum sphere S' enclosing the point set E is computed using an exact solver, such as Gärtner's algorithm (Line 3). After this, a final iteration over all the points in P makes sure that the sphere is grown if necessary to include all points. Each time a point outside the current sphere is encountered, a new larger sphere enclosing the current sphere and the point is computed (Line 4). It is of course expected that $n > k$ from the beginning, otherwise there is no need to reduce the number of points passed to the exact solver (Lines 1, 5, 6). Since the minimum sphere of E is always computed, the resulting method is called the Extremal Points Optimal Sphere (EPOS) algorithm. Given that the points in E include the actual points of support of the optimal sphere, the algorithm is guaranteed to compute the minimum sphere.

Clearly, it is important that the extremal points in E are selected wisely. In addition, since speed is a major concern here, the selection method must be highly efficient as well. For these reasons, the extremal points E are selected by using a pre-determined normal set N with $s = k/2$ normals. For each normal, two extremal points are selected based on projecting all the input points $\mathbf{p}_i \in P$ on the normal, which is illustrated in Figure 2.

[*] e-mail: thomas.larsson@mdh.se

type	#op ($\mathbf{p} \cdot \mathbf{n}$)	#normals	normals
0 0 1	0	3	(1, 0, 0), (0, 1, 0), (0, 0, 1)
1 1 1	2	4	(1, 1, 1), (1, 1, -1), (1, -1, 1), (1, -1, -1)
0 1 1	1	6	(1, 1, 0), (1, -1, 0), (1, 0, 1), (1, 0, -1), (0, 1, 1), (0, 1, -1)
0 1 2	2	12	(0, 1, 2), (0, 2, 1), (1, 0, 2), (2, 0, 1), (1, 2, 0), (2, 1, 0) (0, 1, -2), (0, 2, -1), (1, 0, -2), (2, 0, -1), (1, -2, 0), (2, -1, 0)
1 1 2	3	12	(1, 1, 2), (2, 1, 1), (1, 2, 1), (1, -1, 2), (1, 1, -2), (1, -1, -2) (2, -1, 1), (2, 1, -1), (2, -1, -1), (1, -2, 1), (1, 2, -1), (1, -2, -1)
1 2 2	4	12	(2, 2, 1), (1, 2, 2), (2, 1, 2), (2, -2, 1), (2, 2, -1), (2, -2, -1) (1, -2, 2), (1, 2, -2), (1, -2, -2), (2, -1, 2), (2, 1, -2), (2, -1, -2)

Table 1: *The 49 normals used for efficient computation of extremal points. Here the normals have been divided into six groups based on the operation cost (add, sub, mul) of performing a simple dot product* $\mathbf{p} \cdot \mathbf{n}$.

```
EXTREMALPOINTSOPTIMALSPHERE($P, N$)
    input: $P = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_n\}$ and $N = \{\mathbf{n}_1, \mathbf{n}_2, ..., \mathbf{n}_s\}$
    output: A bounding sphere $S = \{\mathbf{c}, r\}$
    if $(n > (k \leftarrow 2s))$ then
        $E \leftarrow$ FINDEXTREMALPOINTS($P, N$)
        $S' \leftarrow$ MINIMUMSPHERE($E$)
        $S \leftarrow$ GROWSPHERE($P, S'$)
    else
        $S \leftarrow$ MINIMUMSPHERE($P$)
```

Figure 1: *Pseudocode for the EPOS algorithm.*

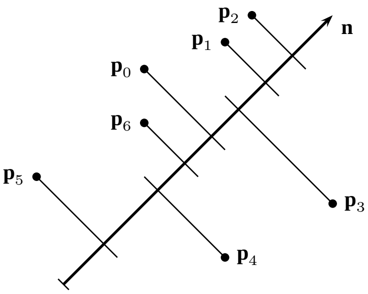


Figure 2: *To select two extremal points along a given normal* $\mathbf{n}$*, the input points* $\mathbf{p}_i \in P$ *are projected on the normal. The points with maximum and minimum projection values,* $\mathbf{p}_2$ *and* $\mathbf{p}_5$ *in this case, are selected as the extremal points.*

By using only integer values, with as many zero and one values as possible for the x, y, and z components of the normals, and still making sure that the normal directions are reasonably uniformly distributed over all possible normals of a hemisphere, the projection of the points using the dot product can be computed more efficiently. The normals used are listed in Table 1. As can be seen, up to 49 normals are used. The cost is measured as the number of arithmetical operations required to compute a dot product using the different types of normals. By unrolling the loop over the normals when projecting the points in P, the unnecessary operations for each computed dot product is easily eliminated. Also, by varying s, the relation between tightness of fit and computational speed is easily controlled. Currently, we have tested specialized implementations of the algorithm using $s \in \{3, 7, 13, 49\}$, which corresponds to selecting $k \in \{6, 14, 26, 98\}$ extremal points in E. The corresponding implementations are hereafter called EPOS-6, EPOS-14, EPOS-26, and EPOS-98.

3 Experimental Results

The experimental evaluation of the EPOS algorithm includes EPOS-6, EPOS-14, EPOS-26, and EPOS-98, as well as five other competing methods. The exact solver by Gärtner with the code on the web is used[1]. The simple but high quality approximation method by Bădoiu-Clarkson, based on core-sets is also used [Bâdoiu and Clarkson 2003] with 100 iterations used for the main loop. Furthermore, three very simple and extremely fast methods are included. These are Ritter's method [Ritter 1990], and the obvious methods of choosing either the mid point of an axis-aligned bounding box (AABB), or the average point, as the center of the sphere. These three methods have in common the fact that they make only two passes over the input points. The algorithms have been implemented in C++, compiled under Microsoft Visual Studio 2005 using the default release mode setting, and run single-threaded on a laptop PC with an Intel T2600 CPU, 2.16 GHz, with 1 GB of memory.

Rather than using random point sets, the data sets used in the benchmarks here are the vertices of 10 polygonal meshes, which is believed to be characterizing for a large number of models used in interactive computer graphics applications. Rendered images of the test models together with their bounding spheres are shown in Figure 3. In Table 2, the results from the benchmarks are presented for all models. All approximation algorithms give significant speedups compared to Gärtner's exact method. In many cases, the approximation methods are two or three orders of magnitudes faster.

The average center method is always the fastest, but it also produces bad fitting spheres in many cases, with a radius of up to twice the length of the minimum. Among the used data sets, the frog model is the worst with 26.36% increase in radius. Ritter's and the AABB center methods are also very fast, but they may also compute loose fitting spheres. For example, consider the tetrahedron model where the increase in radius is 23.28% and 26.93%, respectively. Ritter's algorithm also computes loose fitting spheres with more than a 10% increase in radius for the frog, chair, and knot models.

The proposed EPOS algorithm, on the other hand, manages to compute tight fitting spheres efficiently. The worst cases encountered over all test models were radius increases of 0.05%, 0.21%, 0.91%, and 8.94% for EPOS-98, EPOS-26, EPOS-14, and EPOS-6, respectively. Also, the last mentioned method has a comparable execution speed to both Ritter's method and the AABB center algorithm, but clearly yields tighter fitting volumes.

Furthermore, with the approximation accuracy used here for the Bădoiu-Clarkson method, it becomes significantly slower than all the tested variants of the EPOS method, and still it computes looser

[1] www.inf.ethz.ch/personal/gaertner/miniball.html

Model: frog, $N_v = 4010$, $N_s = 3$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.59903	0.00	207.65	1.00
Bădoiu-Clarkson	0.59997	0.16	4.73	43.91
EPOS-98	0.59903	0.00	1.42	146.35
EPOS-26	0.59903	0.00	0.36	572.21
EPOS-14	0.60019	0.19	0.22	926.80
EPOS-6	0.61349	2.41	0.12	1804.11
AABB center	0.63922	6.71	0.10	2053.30
Ritter	0.65965	10.12	0.10	2058.99
Average center	0.75696	26.36	0.07	2892.20

Model: golfball, $N_v = 100722$, $N_s = 4$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.50110	0.00	4343.12	1.00
Bădoiu-Clarkson	0.50216	0.21	116.98	37.13
EPOS-98	0.50112	0.00	29.51	147.16
EPOS-26	0.50114	0.01	7.56	574.18
EPOS-14	0.50145	0.07	4.80	903.91
EPOS-6	0.50155	0.09	2.52	1720.50
AABB center	0.50197	0.17	2.37	1830.28
Ritter	0.51531	2.83	2.27	1916.47
Average center	0.50189	0.16	1.72	2518.04

Model: horse, $N_v = 48485$, $N_s = 2$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.62897	0.00	1128.45	1.00
Bădoiu-Clarkson	0.62897	0.00	56.26	20.06
EPOS-98	0.62897	0.00	14.43	78.18
EPOS-26	0.62897	0.00	3.72	303.60
EPOS-14	0.62899	0.00	2.34	481.96
EPOS-6	0.63023	0.20	1.23	918.03
AABB center	0.63517	0.99	1.15	979.71
Ritter	0.63476	0.92	1.11	1017.47
Average center	0.65200	3.66	0.83	1359.13

Model: hand, $N_v = 327323$, $N_s = 2$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.52948	0.00	6086.75	1.00
Bădoiu-Clarkson	0.52992	0.08	390.04	15.61
EPOS-98	0.52948	0.00	95.94	63.44
EPOS-26	0.52949	0.00	24.78	245.60
EPOS-14	0.52949	0.00	15.81	385.02
EPOS-6	0.52949	0.00	8.38	726.53
AABB center	0.53029	0.15	7.93	767.26
Ritter	0.52949	0.00	7.60	800.43
Average center	0.61943	16.99	5.94	1024.20

Model: bunny, $N_v = 32875$, $N_s = 3$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.64321	0.00	1680.20	1.00
Bădoiu-Clarkson	0.64792	0.73	38.07	44.13
EPOS-98	0.64328	0.01	9.69	173.36
EPOS-26	0.64415	0.15	2.49	675.77
EPOS-14	0.64423	0.16	1.58	1062.04
EPOS-6	0.65017	1.08	0.83	2029.13
AABB center	0.67296	4.63	0.78	2164.99
Ritter	0.67694	5.24	0.74	2285.09
Average center	0.74940	16.51	0.56	2992.21

Model: tetrahedron, $N_v = 32770$, $N_s = 4$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.61237	0.00	3361.89	1.00
Bădoiu-Clarkson	0.61237	0.00	37.90	88.71
EPOS-98	0.61237	0.00	9.71	346.26
EPOS-26	0.61237	0.00	2.49	1349.41
EPOS-14	0.61237	0.00	1.58	2123.15
EPOS-6	0.61237	0.00	0.83	4054.59
AABB center	0.77728	26.93	0.78	4331.90
Ritter	0.75494	23.28	0.74	4522.37
Average center	0.61238	0.00	0.56	5990.06

Model: teapot, $N_v = 32922$, $N_s = 3$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.50313	0.00	2043.18	1.00
Bădoiu-Clarkson	0.50314	0.00	38.70	52.79
EPOS-98	0.50322	0.02	9.88	206.76
EPOS-26	0.50322	0.02	2.54	805.91
EPOS-14	0.50322	0.02	1.60	1274.82
EPOS-6	0.50322	0.02	0.83	2450.14
AABB center	0.51913	3.18	0.78	2612.96
Ritter	0.50322	0.02	0.74	2761.96
Average center	0.54038	7.40	0.57	3581.62

Model: chair, $N_v = 7260$, $N_s = 4$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.63776	0.00	482.73	1.00
Bădoiu-Clarkson	0.63777	0.00	8.45	57.10
EPOS-98	0.63788	0.02	2.30	209.50
EPOS-26	0.63792	0.03	0.59	819.32
EPOS-14	0.64359	0.91	0.37	1288.55
EPOS-6	0.69474	8.94	0.19	2482.68
AABB center	0.64737	1.51	0.18	2747.14
Ritter	0.73014	14.49	0.17	2870.35
Average center	0.73279	14.90	0.13	3814.45

Model: knot, $N_v = 1440$, $N_s = 4$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.54180	0.00	5.69	1.00
Bădoiu-Clarkson	0.54372	0.35	1.68	3.39
EPOS-98	0.54193	0.02	0.64	8.90
EPOS-26	0.54262	0.15	0.16	36.24
EPOS-14	0.54259	0.14	0.09	61.72
EPOS-6	0.54382	0.37	0.05	122.69
AABB center	0.59055	9.00	0.04	151.99
Ritter	0.61211	12.98	0.04	151.99
Average center	0.54181	0.00	0.03	216.66

Model: tiger, $N_v = 30892$, $N_s = 3$				
Algorithm	radius	inc(%)	time(ms)	speed-up
Gärtner	0.51397	0.00	238.16	1.00
Bădoiu-Clarkson	0.51704	0.60	35.74	6.66
EPOS-98	0.51424	0.05	9.11	26.15
EPOS-26	0.51507	0.21	2.34	101.85
EPOS-14	0.51507	0.21	1.48	160.55
EPOS-6	0.52531	2.21	0.78	306.44
AABB center	0.56327	9.59	0.73	327.26
Ritter	0.53835	4.74	0.78	306.77
Average center	0.55602	8.18	0.53	450.59

Table 2: *Experimental results obtained for 10 models (polygonal meshes) with N_v vertices. N_s is the number of supporting points for the minimum enclosing ball. For each algorithm, the resulting radius and execution time are given. For convenience, the increase in radius and speed-up compared to the exact solver by Gärtner are also given.*

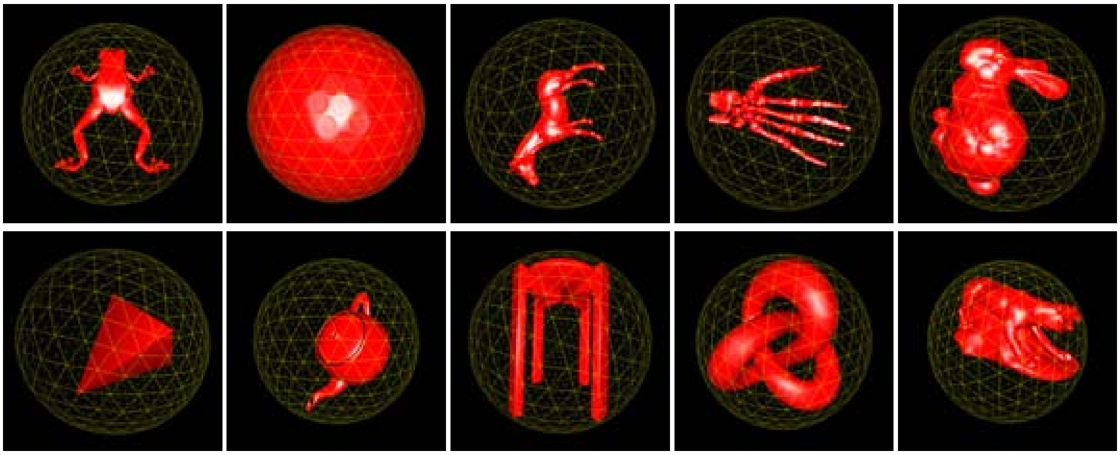


Figure 3: *Visualization of the used models. The bounding spheres shown here in wireframe were computed using the EPOS-26 algorithm.*

fitting spheres than e.g. EPOS-98 and EPOS-26 for six out of 10 test models (frog, golfball, hand, bunny, knot, and tiger).

4 Conclusions and Future Work

As shown by the experimental results, there is a huge performance gap between exact solvers and approximation methods. Non-optimal bounding spheres can be computed several orders of magnitude faster than minimum spheres. Therefore, their use is widespread in time-critical situations. On the other hand, in many applications bounding spheres can be pre-computed. In this case, minimum bounding spheres are often preferable.

The proposed EPOS algorithm is fast, robust, and it computes high-quality spheres. Therefore, it may be suitable for real-time graphics and visualization applications. For example, the EPOS-6 algorithm can with advantage replace Ritter's method in many cases, since in general it computes tighter fitting spheres at roughly the same speed. In other interactive contexts, the EPOS-14, EPOS-26, or even the EPOS-98 may be applicable, since they clearly produce tighter fitting spheres. However, they are approximately 2, 3, and 12 times slower than the EPOS-6, respectively, in all the experiments.

Interesting future work includes designing a parallel version of the EPOS algorithm which utilizes loop parallelization methods on many core architectures, as well as modern vectorized instructions sets, such as Intel's SIMD SSE. Furthermore, computing the smallest enclosing ball of balls is a similar problem, and it would be interesting to modify the EPOS algorithm to handle this case as well [Fischer and Gärtner 2003].

Also, since the EPOS algorithm generalizes to the n-dimensional case, experimental evaluation for other dimensions than 3D would be interesting. In 2D, the EPOS algorithm can be used with advantage to solve the smallest enclosing circle problem. Good results are expected by using the following fixed normal set when selecting extremal points: (0, 1), (1, 0), (1, 1), (1, -1), (1, 2), (2, 1), (2, -1), and (1, -2). In dimensions $d \geq 4$, a procedure for automatic generation of appropriate normals would be preferable. Unfortunately, the number of needed normals to compute tight fitting balls grows superlinearly with the dimension d.

Finally, the EPOS algorithm can be modified to dynamically determine a more advantageous normal set by taking the actual distribution of the input points into consideration (cf. [Wu 1992]). This may be worthwhile if fewer normals can be selected to reach the same quality of the computed spheres as in the present solution.

References

BÂDOIU, M., AND CLARKSON, K. L. 2003. Smaller core-sets for balls. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 801–802.

BĂDOIU, M., AND CLARKSON, K. L. 2008. Optimal core-sets for balls. *Computational Geometry: Theory and Applications 40*, 1, 14–22.

EBERLY, D. H. 2007. *3D Game Engine Design: A Practical Approach to Real-Time Computer Graphics, Second Edition*. Morgan Kaufmann.

ERICSON, C. 2005. *Real-Time Collision Detection*. Morgan Kaufmann.

FISCHER, K., AND GÄRTNER, B. 2003. The smallest enclosing ball of balls: combinatorial structure and algorithms. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry*, ACM, New York, NY, USA, 292–301.

GÄRTNER, B. 1999. Fast and robust smallest enclosing balls. In *ESA '99: Proceedings of the 7th Annual European Symposium on Algorithms*, Springer-Verlag, London, UK, 325–338.

KUMAR, P., MITCHELL, J. S. B., AND YILDIRIM, E. A. 2003. Approximate minimum enclosing balls in high dimensions using core-sets. *Journal of Experimental Algorithmics 8*.

MEGIDDO, N. 1983. Linear-time algorithms for linear programming in R^3 and related problems. *SIAM Journal on Computing 12*, 759–776.

RITTER, J. 1990. An efficient bounding sphere. In *Graphics Gems*, A. Glassner, Ed. Academic Press, 301–303.

WELZL, E. 1991. Smallest enclosing disks (balls and ellipsoids). In *New Results and Trends in Computer Science, Lecture Notes in Computer Science 555*, H. Maurer, Ed. Springer, 359–370.

WU, X. 1992. A linear-time simple bounding volume algorithm. In *Graphics Gems III*, D. Kirk, Ed. Academic Press, 301–306.

Real Time Large Scale Fluids for Games

Daniel Kallin*

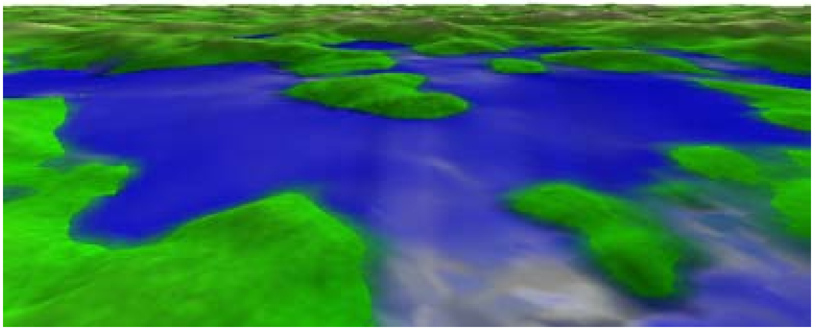


Figure 1: A virtual sea

Abstract

This paper presents an implementation of a stable height-field fluid solver on non-uniform quadtree grids. Smoothing kernel interpolation allow one to use semi-Lagrangian advection on the non-uniform grid. A modification to the advection make it mass conserving. The non-uniform grid allow the model to run on a very tight cell budget with high frame rates even for arbitrary sized environments. Gravity acceleration is implemented as a modified explicit Euler step with upwind differencing. The differencing is based on a definition of a node neighbour which doesn't require the neighbor to be a quadtree leaf. This solves the problems of hanging nodes inherent with quadtrees. This model is suitable for water propagating over height field terrains in interactive environments like video games.

Keywords: fluid dynamics, semi-Lagrangian advection, upwind differencing, adaptive grid, quadtree

1 Introduction

This paper describes the research into faster than real time fluid simulations done as a master thesis project at Avalanche Studios, an independent video game developer in Stockholm, Sweden. The premise for the research was the special set of demands that contemporary video games have. It is the desire to improve the interactivity of the game environment that pushes the technological boundaries. At the same time the physical accuracy of the simulations are of lower priority than the interactivity and the plausibility of the environment as perceived by a casual player. The speed requirements are very strict while the demands on the algoritm are very high with complex boundary conditions consisting of both fixed boundaries at the terrain-fluid interface and dynamic boundaries at the air-fluid interface.

2 Previous work

The field of computational fluid dynamics has been extensively researched, primarily for engineering purposes but also extensively by the graphics and animation community for visualization purposes. While it is beyond the scope of this report to present an exhaustive breakdown of the field some key publications which has inspired this research will be briefly presented.

2.1 Navier Stokes integration

On a full 3D grid the Navier Stokes equations can be discretized and solved numerically, an approach which can capture all the phenomena of real world fluids of longer wavelength than the grid cell size.

Jos Stam [Stam 2003] show an implementation of the Navier Stokes equations that is unconditionally stable with time steps of arbitrary lengths. It uses implicit diffusion and a combination of particle and grid based implicit advection.

2.2 Height field simulation

When not using uniform isotropic grids one can reduce the complexity of the model. Shallow lakes and puddles are examples of situations where a uniform isotropic grid could be replaced by a height field model where there is no grid subdivisioning along one axis.

Kass and Miller [Kass and Miller 1990] describe an implementation of a height field fluid based on a simplified set of the Shallow Water equations. Their integrator use alternating direction differencing with implicit integration. Their simplified model does not capture the advection of velocity through the fluid domain. It can handle dynamic boundaries of the fluid volume which means that their fluid can move across dry terrain and recede from flooded areas.

*e-mail: daniel.kallin@gmail.com

Maes, Fujimoto and Chiba [Maes et al. 2006] describe a height field model where the columns are subdivided to capture vertical inhomogenities of the fluid. Their simulations however only show minor quality gains from this increase in resolution.

2.3 Particle based simulation

Smoothed Particle Hydrodynamics (SPH) has since its introduction to the graphics community attracted attention and Hegeman, Carr and Miller [Hegeman et al. 2006] show how the simulation can be processed on the GPU. Even if SPH models have cubic complexity when filling volumes the general problem is not the simulation but rather the subsequent visualization of the data [Mller-Fischer 2007].

Müller, Schirm and Duthaler [Müller et al. 2007] show how the visualization of particle fluids can be done in screen space for increased performance.

2.4 Hybrid approaches

Irving, Guendelman, Losasso and Fedkiw [Irving et al. 2006] describe a combination of a height field and a 3D grid to very accurately model the free surface of water with arbitrary topology while maintaining fast simulation speeds. Their implementation is as such adaptive in the up direction but not adaptive along any other axes. This makes this approach slow for fluid volumes that cover large areas but the visual quality of their simulation is very good.

Cords [Cords 2007] describe an application of mode-splitting where the volume filling flow is simulated using smoothed particle hydrodynamics and the high frequency surface waves are simulated as a 2D wave equation superimposed on the surface generated from the SPH simulation. While this approach yields extremely attractive water surfaces the wave equation must still be solved for a high resolution grid.

3 Theory

3.1 Navier-Stokes equations

The Navier-Stokes equations governing the dynamics of incompressible fluids can be stated as [Hoffman and Johnson 2006]:

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} + \nu \nabla^2 \mathbf{u} + \mathbf{f}$$

$$\nabla \cdot \mathbf{u} = 0$$

These equations describe the behavior of the velocity field $\mathbf{u}$ with the acceleration $\frac{\partial \mathbf{u}}{\partial t}$ as a sum of three terms. The first term $-(\mathbf{u} \cdot \nabla)\mathbf{u}$ describes advection: the inertial transport of velocity with the fluid flow. The second term $\nu\nabla^2\mathbf{u}$ models the viscosity of the fluid as diffusion of the velocity field. $\mathbf{f}$ is the sum of all applied forces such as gravity or forced flows. The second equation $\nabla \cdot \mathbf{u} = 0$ represents the requirement of the fluid to be incompressible.

3.2 Shallow Water Equations

The Shallow Water Equations are derived from the Navier-Stokes equations with certain assumptions and approximations made: they apply to thin, non turbulent layers of fluid which acts as a height field [Wong 2003]. These equations have been used extensively in graphics applications [Kass and Miller 1990; Thurey et al. 2007] due to their simplicity and speed - the assumption of 2D reduces the worst-case complexity of the simulation to $\mathcal{O}(n^2)$ instead of $\mathcal{O}(n^3)$ which applies to the full Navier Stokes 3D-grid simulations.

As stated in Bridson and Müllers SIGGRAPH course on fluid simulation [Bridson and Mller-Fischer 2006] the numerical dissipation is similar to but dominant over the real world viscous dissipation for most fluids which means that the viscosity term often can be ignored. However solvers that attempt to operate at interactive frame rates must take large time steps Δt which negatively affects the stability of the model. Imposing explicit non-physical dissipation in these cases would increases stability.

Below the modified shallow water equations with diffusion and viscosity implemented is presented:

$$\frac{\partial h}{\partial t} = -\nabla \mathbf{u} d + \kappa \nabla^2 h$$

$$\frac{\partial \mathbf{u}}{\partial t} = -g\nabla h - (\mathbf{u} \cdot \nabla)\mathbf{u} + \nu \nabla^2 \mathbf{u}$$

These equations model the change in height $\frac{\partial h}{\partial t}$ as a function of the velocity $\mathbf{u}$, the depth d (defined as the difference between the water height h and the height of the sea floor) and a diffusion term $\kappa\nabla^2 h$. The acceleration of the velocity field is modelled as a sum of the acceleration due to gravity $-g\nabla h$, an advection term and a viscosity term $\nu\nabla^2\mathbf{u}$.

3.3 Eulerian and Lagrangian models

Eulerian models use discretized grids that remain fixed in the frame of reference. The individual cells communicate with each other and phenomena are propagated by transmissions across cell boundaries. Discrete versions of the Navier-Stokes or the Shallow Water equations are integrated using a solver of one's choice. These models are beneficial since they put the designer in control over the resolution. Their regularity also allow for easy visualization and quick isosurface extraction when simulating the fluid as a 2.5D height field [Kass and Miller 1990]. Eulerian models are rather poor at modelling advection and energy and mass conservation. Special care must be taken to guarantee constant volume in the fluid either in the solver itself or as a post processing step. Even worse is that an inappropriate choice of integrator for the advection will lead to severe instabilities when the time step is lengthened, something which today is required when attempting to attain real time speeds.

Lagrangian particle models model interactions in a moving frame of reference. The fluid is represented by a multitude of particles which interact with each other and their surroundings. This model is by its nature mass conserving since every quantum of fluid is represented by a particle and is thus accounted for by the model. As a drawback the model can invest a large amount of memory and computing power in volumes that are filled but might be uninteresting to an observer. The visualization of the fluid is also problematic since the generation of an isosurface is more expensive when dealing with the randomly distributed particles [Green 2008]. Hugoson and Nilsson describes an approach of surface normal blending which eschews the generation of a geometric isosurface altogether [Hugoson and Nilsson 2007].

3.4 Explicit integrations

As noted by Kass and Miller [Kass and Miller 1990] explicit integrators such as Euler's method are a poor choice when solving the Shallow Water equations. When the wave velocity approaches one grid cell per time step the simulation tend to diverge. The time step Δt and gravity g must be strictly limited to avoid oscillations and divergent behavior [Bridson et al.].

To maintain stable simulations implicit solvers are desired. The Implicit Euler method attempt to find a state which when run backwards in time using the explicit method yield the initial state. As noted by Stam [Stam 2003] this approach is problematic since the fluid is advected by the velocity field which in turn is self-advected. A possible approach is to use different solvers for different terms in the equations. Stam describes how one can integrate the advection using a Lagrangian method and the diffusion using an implicit Eulerian method with Gauss Seidel relaxation.

3.5 Semi-Lagrangian advection

A hybrid approach which has been embraced by the interactive computational fluid dynamics community is the semi-Lagrangian model [Stam 1999]. It represents the fluid using a grid and models the dynamics using particles. All attributes such as velocity, density and pressure are bound to the grid but their advection by the velocity field is not handled by cell to cell communications. Instead, mass-less particles follows the velocity field and transports the attributes from their departure points to their destinations after a single time step Δt. This tracing can be performed forward and backwards in time, yielding the forward and backward semi-Lagrangian model.

In the backward semi-Lagrangian model the particles' destinations are set as the centers of the grid cells and their departure points are calculated from the velocity field. The particles carry with them the attributes from their departure points which does not necessarily line up with the grid. At this point one must therefore extract an interpolated value from the grid [Bridson et al.].

4 Techniques

The model used in the prototype software is a height field fluid simulation on a 2.5D terrain. Both the fluid and the terrain data reside in the same quad tree data structure where every node in the tree contains a complete set of data. Non-leaf nodes have values that are averages of its four child nodes' values. The fluid is simulated using a modified set of the Shallow Water Equations where the advection is handled using semi-Lagrangian advection and the influence of gravity, friction and diffusion are modelled using slightly modified explicit Euler integrators.

4.1 Gravity acceleration

The Shallow Water Equations (see Section 3.2) model the acceleration due to gravity as $\frac{\partial \mathbf{u}}{\partial t} = -g\nabla h$. The prototype implementation the gravity acceleration is explicitly modelled. This led to severe artifacts such as introduced oscillations caused by the inherent properties of explicit Euler integrators. While a solution to this problem would be to use implicit Euler integration for the acceleration this approach was rejected as it would result in an iterating implementation with all of the associated complexity. An alternative would be to keep the explicit Euler method but instead reduce the time step exclusively for the gravity acceleration for thin sheet of fluid where the overshoot was dominant over the expected behaviour of the fluid. The acceleration due to gravity is thus modelled as:

$$\frac{\partial \mathbf{u}}{\partial t} = -\frac{g\nabla h}{\max(1, d)}$$

In the prototype it is observed that without this modification the model exhibit spontaneous oscillations when the cell size is smaller than the depth.

4.2 Upwind differencing

An important part of the model is the upwind differencing which is used in the derivative approximation in the discretized grid [Fitzpatrick] [Rogers and Kwak 1991]. Early prototypes clearly showed that central differences resulted in high frequency noise and undampened oscillations.

The upwind differencing scheme defines the gradient of the height field h as a finite difference whose direction is dependant on the velocity vector $\mathbf{u} = (u, v)$.

$$\nabla h = \left(\frac{\partial h}{\partial x}, \frac{\partial h}{\partial y}\right)$$

$$\nabla h \approx \left(\frac{\Delta^x h_{ij}}{\Delta x}, \frac{\Delta^y h_{i,j}}{\Delta y}\right)$$

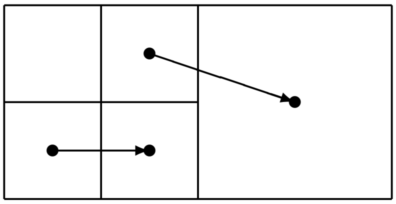
$$\Delta^x h_{i,j} \equiv \begin{cases} h_{i,j} - h_{i-1,j} & u \geq 1 \\ h_{i+1,j} - h_{i,j} & u < 1 \end{cases}$$

$$\Delta^y h_{i,j} \equiv \begin{cases} h_{i,j} - h_{i,j-1} & v \geq 1 \\ h_{i,j+1} - h_{i,j} & v < 1 \end{cases}$$

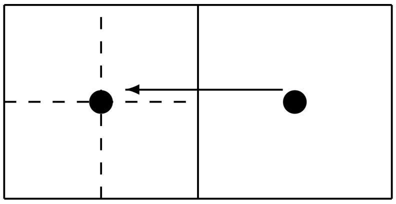
4.3 Adaptive grids

In contemporary video games dynamic level of detail adjustment is employed to optimize the complexity of game elements such as geometry to ensure high frame rates [Luebke et al. 2003]. As opposed to individual models which can be up or downsampled individually a flooded terrain is a very large and potentially continuous piece of geometry which would require different levels of detail in different areas [Duchaineau et al. 1997]. Computational fluid dynamics is demanding and for this reason it is desired to use different resolutions in different areas of the grid [Mller-Fischer 2007]. The required level of detail is dependent on the static terrain through which the fluid flows, the actual distribution of fluid and its velocity and the position of the observer. Apart from the terrain all of these are dynamic factors which would require us to dynamically change the resolution of the grid. It is beneficial to base the grid on a quadtree, since they allow for arbitrary subdivision of cells without affecting the surrounding cells thus simplifying dynamic resolution changes. The grid cells are also all square and symmetric further simplifying the discretization of the underlying equations. A quadtree can potentially possess hanging nodes, something which more complex subdivisioning techniques such as ROAM [Duchaineau et al. 1997] lack. This means that situations can arise where a single edge of a cell can touch several smaller neighboring cells. When discretizing the equations ambiguities arise in how one should formulate the differences approximating derivatives for a large cell that have more than one neighbor on each side. In the explicit gravity acceleration integrator this is solved by forcing cells to never deal with neighbors that are smaller than themselves. Instead they consider the quadtree node that shares the same level as itself to be its neighbor. This non-leaf cell's mass and velocity are calculated as the recursive average of its children's attributes. Smaller cells do not have this ambiguity since they have a single - albeit larger - neighbor.

As a precomputing step each node in the quadtree (on all levels) is assigned four neighbors at their own or a higher level (see figure



(a) Right neighbors



(b) Left neighbor

Figure 2: The neighbors of the quadtree nodes are defined as being at least as large as the node itself. (a): Two cells with arrows pointing to their right-hand neighbors. (b): One cell with arrow pointing to its left-hand neighbor which is a parent of four smaller nodes.

2). These neighbor pointers are used when computing differences for the discretized fluid equations. The actual simulation is only conducted in the leaf nodes but after all leafs have been traversed and simulated, averages are computed from the bottom up to fill every node in the tree with updated values. These averaged values are then returned when a neighbor which isn't a leaf node is polled for its properties (like velocity or fluid height).

Each cell also contains flags that decide if that particular node and its children require updates. All nodes are in the dry initial state set to not require updates and when fluid is added the flags of the filled cells and their neighbors are recursively from the bottom up set to require updates.

In the early prototypes it was observed that explicit methods with parameters that give stable simulations on an uniform grid can produce instabilities in heterogeneous grids in the boundaries between high and low resolution. This hints at a demand for implicit methods that are stable.

4.4 Smoothing kernel interpolation

When implementing semi-Lagrangian advection neighboring cells do not affect each other by virtue of them being neighbors. Instead advection is handled by the transport of mass and velocity properties from the departure point to the arrival point of a massless tracer particle. At the departure point an interpolated value must be extracted from the grid. For uniform grids this can be implemented as a simple bilinear interpolation with cell values defined at a single point somewhere on the cell (usually in the center) [Stam 2003]. On a non-uniform grid however it is not obvious how one extracts an interpolated value from an arbitrary point in an adaptive grid when one is near the interface between high and low resolution (see Fig 3).

The chosen approach was to define the cell properties to be constant over the entire area covered by the cell. This yields a completely uninterpolated grid with possible discontinuities at every cell boundary, which is unacceptable. But on this piecewise constant field values can be interpolated using a smoothing kernel that returns a weighted integral $\hat{f}$ of the surrounding cells' values. This approach allows to both read and write values to an adaptive grid at arbitrary positions with correct averaging of values as defined by the kernel function K.

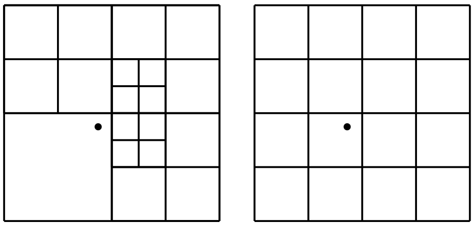


Figure 3: Finding the interpolated value at the dot is non trivial in a quad tree as opposed to a regular grid where bilinear interpolation is sufficient.

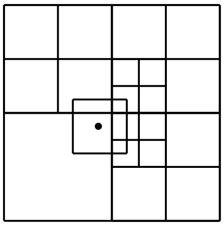


Figure 4: The value at the dot is extracted by using the cells relative coverage of the window as weights in a weighted interpolation of their values.

A uniform square kernel with constant weight within its coverage defined by the bandwidth h can be defined as:

$$\hat{f}(x,y) = \frac{1}{h^2}\int_{\mathbb{R}^2} K\left(\frac{x-x'}{h}, \frac{y-y'}{h}\right) f(x',y')dx'dy'$$

$$K(x,y) = \left\{ \begin{array}{ll} \frac{1}{4} & |x|,|y| < 1 \\ 0 & otherwise \end{array} \right.$$

A central issue when using smoothing kernels is to find an optimal value for the bandwidth h. In quad tree based adaptive grids with arbitrary large difference between cell sizes any constant choice of bandwidth could result in over or under smoothing at some points on the grid. The solution — and a crucial part of the implementation of the semi-Lagrangian method — is to let the bandwidth h of the kernel function depend on the size of the cell which the tracer particle arrives at. In the semi Lagrangian approach each tracer particle belongs to an individual cell with the size s. The kernel function used by the particle at its departure point is defined with a bandwidth $h = \frac{s}{2}$. With uniform kernel functions this corresponds to a square window with sides of length s. The extent to which this window covers the underlying cells is used as a weight when interpolating the properties of all the underlying cells (see Fig. 4). For uniform grids this approach is equivalent to bilinear interpolation, which is a requirement since the algorithm must conform with existing implementations when fed with a uniform grid.

4.5 Mass conserving advection

When modelling a three dimensional incompressible fluid as a height field the model is equivalent to a compressible two dimen-

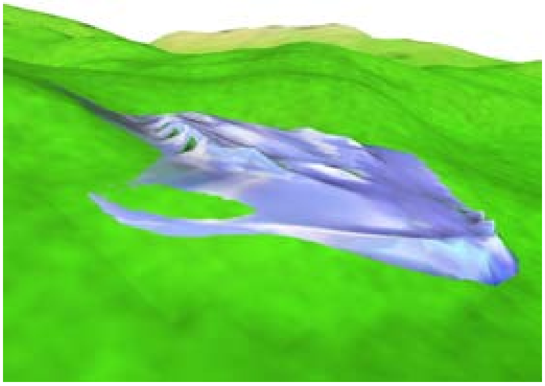


Figure 5: A mass of water flowing down a slope.

sional fluid. Compressibility is required for simulating waves in the model but it causes the velocity fields to no longer be divergence free. Therefore Stam's method to decompose the velocity field and forcing it to be divergence free can not be applied.

The solution was to modify the semi-Lagrangian advection model. Originally it only passively reads values at the tracer particle's departure point and replaces the values at the arrival point. The idea is that the values of the cells surrounding the departure point will attain new values from their tracer particles departure points. This leads to non-physical behavior in a compressible fluid. Instead the model actually subtracts fluid volume from the departure point and add this volume to the arrival point. This subtraction relies on kernel approach for quad tree interpolation as the amount of fluid to be subtracted from each of the surrounding cells is weighted by the kernel function.

4.6 Diffusion

Introducing diffusion into a model is a simple and effective way to increase the stability of the simulation. It is employed in the implementation to dampen high frequency oscillations which arise with long time steps. Issues with volume conservation is encountered when applying diffusion to the fluid height property. Because of this only the velocities are diffused and not the fluid heights. This is motivated by the implicit advection and the upwind differencing scheme which introduce significant damping to the model. A central feature of the model is that the velocities are not only diffused inside the fluid domain but across the entire grid. No diffusion is added to the fluid mass which is only indirectly affected by the velocity diffusion and therefore keeps within its bounds. The velocity however is allowed to diffuse beyond the bounds of the fluid mass and is treated as a virtual velocity present in dry terrain. This virtual velocity is used in the advection step and is essentially what allows the implicit advection routine to propagate fluid out onto dry terrain. Below the definition of the viscous contribution to the velocity derivative and the discretized explicit interpretation is presented:

$$\frac{\partial \mathbf{u}}{\partial t}_{diff} = \nu \nabla^2 \mathbf{u}$$

$$\mathbf{u}_{n+1} = \Delta t \nu \nabla^2 \mathbf{u}_n$$

4.7 Suppression of thin flows

The flow model presented so far contains no mechanism for simulating the friction between fluid and ground. For large flows where most of the water glides on the bottom-most layer the friction is negligible but for thin and fast flows the friction becomes dominant since fluid friction scales quadratically with the flow velocity.

$$\mathbf{F} \propto \rho \mathbf{u}^2$$

The modelling of this phenomenon is further complicated by the effects of absorption when small volumes of water would rather penetrate the ground than flow on top of it. Several different approximations of friction with the initial approach incorporating the absorption effect was tested.

One approach was to specify a minimum depth d_{min} and for cells with shallower depths than this threshold forcing an artificial decay of the fluid velocity. Small amounts of fluid would then be locked to terrain instead of flowing over it freely approximating the absorption. In the subsequent visualization the drawing of fluid layers shallower than d_{min} was omitted. This approach gave visually satisfactory results but produced artifacts in the behavior of wavefront propagating over dry terrain.

Instead a proper fluid velocity decay was derived from the equations of fluid friction. The frictional force per unit area is $F = -ku^2$ in the direction of flow $\hat{\mathbf{u}}$. In the height field model all properties including forces are treated as being constant along the entire column heights and thus the mass this force acts on is proportional to the depth. The resulting frictional acceleration is $\mathbf{a} = -\frac{ku^2}{d}\hat{\mathbf{u}}$. With forward Euler integration one obtains $\mathbf{u_{n+1}} = \mathbf{u_n} + \Delta t\mathbf{a}$. An observation is that long time steps can result in frictional forces actually reversing the flow, a behavior which is clearly unphysical and due to the shortcomings of the forward Euler method. Similarly to the treatment of the gravitational acceleration the frictional force is clipped at the value where it passes zero. With this modification and by transforming the frictional addition to a multiplicative factor one yield:

$$\mathbf{u_{n+1}} = \mathbf{u_n} \max\left(0, 1 - \Delta t \frac{kv}{d}\right)$$

5 Results

Since the method uses a non-uniform grid that is only traversed where it contains fluid it is meaningless to talk about performance as related to grid size. Benchmarking shows that there is no practical performance hit of having inactive (dry) areas in the grid making it feasible to let the grid cover the entire virtual environment in anticipation of added fluid. After the fluid has passed over an area it would be possible to deactivate the cells and even dynamically reshape the quadtree when the observer moves across the terrain. Because of this the performance is presented as the time cost per active cell. The approximate time per iteration per active cell for a 2 GHz Athlon64 test computer were 1.7 microseconds. It should be noted that the implemenation could be further optimized and that the model as such allow for computational grids which contain very few grid cells.

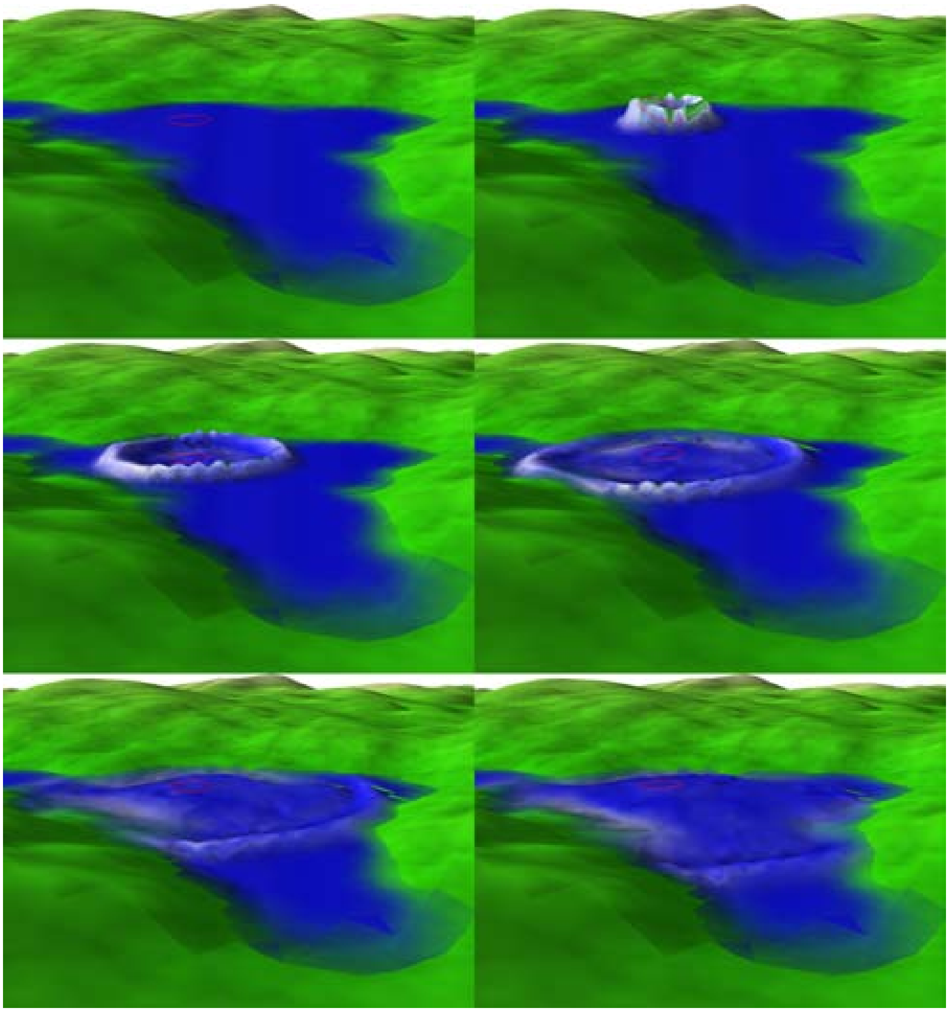


Figure 6: A wave spreading in the prototype software. Wave is spawned at the position marked by the red circle.

6 Discussion

6.1 Design choices in the model

While the modified shallow water equations can be integrated in a unified explicit step the approach described in this report have divided the fluid model into tree separate routines: dissipation, gravity acceleration and advection. These routines are performed in a simulation pass that calculates a new state from the old one. This is followed by a separate update pass that sets the current state as the new state.

6.1.1 Explicit dissipation

The dissipation is explicitly modelled. While this leads to instabilities with long time steps and heavy dissipation the explicit dissipation can be considered a sensible choice since the target fluid - water - has a low natural viscosity. Therefore the parameters are still small enough to stay within the bounds of stability and if one wishes to model a more viscous fluid there are other approaches that are more suitable models than the one presented here.

Problems with mass conservation arose along fluid boundaries when diffusing the fluid mass itself and therefore the model only applies diffusion to the velocity field. Greater stability might be attained by diffusing the height field as well with the possibility of using a smaller diffusion parameter. The reader is invited to examine this problem of efficient volume conserving diffusion.

6.1.2 Explicit gravity acceleration

The gravity acceleration step is the weakest point of the model. The acceleration is clipped on large water depths (see section 4). At larger time steps it promptly introduces instabilities in small cells, as expected from an explicit routine. The reader is advised to substitute this approach for a more stable implicit routine if larger time steps is a requirement.

6.1.3 Advection

The model's advection routine is a mass conserving semi Lagrangian method with smoothing kernel interpolation on non-uniform grids. The fact that each advection operation in itself is mass conserving with subtraction at one point and addition at another makes it possible to skip the expensive conservation passes employed by Stam [Stam 2003] and others. The smoothing kernel approach is essential for implementing semi Lagrangian advection on adaptive grids. The downside is that there is a lot of overhead associated with the advection routine. Especially the interpolation step is expensive in the prototype. Twice per cell and iteration the quadtree is traversed from the root to the leaves where every cell that is covered by the kernel function is modified. It should be feasible to use the presented approach without starting the search from the root. It is theoretically possible but in practice rare that the departure point of the tracer particle lies far away from the arrival point. It might be worthwhile to research optimized advection routines that still uses kernel interpolation.

6.1.4 Other applications

The core idea of this paper is the use of a semi-Lagrangian approach on an adaptive grids using a kernel function to read and write values to and from the grid. This approach should be applicable to other problems than just the Shallow Water equations. On a three dimensional adaptive grid we can use the same approach to solve the Navier Stokes equations. Other problems than fluid dynamics might also be suitable for simulation using a semi-Lagrangian model with non uniform grid.

7 Conclusion

The method presented in this report have certain shortcomings but it presents valuable solutions which should be readily applicable to other methods. It is the author's conclusion that real time fluid dynamics in computer games is possible. But it is not unconditionally possible. It is associated with a significant computational cost and — more importantly — a significant investment in research.

The research is not necessarily about physics or computation. An important question is what we humans associate with realistic fluid behaviour. With such research we might be able to develop routines to simulate these perceived fluid behaviours.

References

ASPLUND, P. 2007. *Dynamic Spreading of Wildfire*. Master's thesis, Kungliga Tekniska Hgskolan.

AXELSSON, T. 2007. *Realtidssimulering av vtskor i datorspel mha partikelsystem*. Master's thesis, Hgskolan Skvde.

BEAUDOIN, P., PAQUET, S., AND POULIN, P. 2001. Realistic and controllable fire simulation. In *Graphics Interface 2001*, 159–166.

BRIDSON, R., AND MLLER-FISCHER, M., 2006. Fluid simulation course notes, siggraph.

BRIDSON, R., AND MLLER-FISCHER, M. 2007. Combating dissipation. SIGGRAPH.

BRIDSON, R., AND MLLER-FISCHER, M. 2007. Fluid simulation. SIGGRAPH.

BRIDSON, R., AND MLLER-FISCHER, M. 2007. More accurate pressure solves. SIGGRAPH.

BRIDSON, R., FEDKIW, R., AND MLLER-FISCHER, M. Fluid simulation course notes, siggraph 2007.

CORDS, H. 2007. Mode-splitting for highly detailed, interactive liquid simulation. In *GRAPHITE '07: Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, ACM, New York, NY, USA, 265–272.

DUCHAINEAU, M. A., WOLINSKY, M., SIGETI, D. E., MILLER, M. C., ALDRICH, C., AND MINEEV-WEINSTEIN, M. B. 1997. ROAMing terrain: real-time optimally adapting meshes. In *IEEE Visualization*, 81–88.

DUNN, A., AND MILNE, G. 2004. Modelling wildfire dynamics via interacting automata. In *Cellular Automata*, Springer Berlin, vol. 3305, 395–404.

ELCOTT, S. 2005. *Discrete, Circulation-Preserving, and Stable Simplicial Fluids*. Master's thesis, California Institute of Technology.

FITZPATRICK, R. Upwind differencing. [Online; accessed 9-June-2008].

FOSTER, N., AND METAXAS, D. 1996. Realistic animation of liquids. *Graphical models and image processing: GMIP 58*, 5, 471–483.

GREEN, S., 2008. Particle-based fluid simulation, nvidia.

HEGEMAN, K., CARR, N. A., AND MILLER, G. S. 2006. Particle-based fluid simulation on the gpu. In *Computational Science – ICCS 2006*, Springer, V. N. Alexandrov, G. D. van Albada, P. M. Sloot, and J. Dongarra, Eds., vol. 3994 of *LNCS*, 228–235.

HINSINGER, D., NEYRET, F., AND CANI, M.-P. 2002. Interactive animation of ocean waves. In *ACM-SIGGRAPH/EG Symposium on Computer Animation (SCA)*.

HOFFMAN, J., AND JOHNSON, C. 2006. *Computational Turbulent Incompressible Flow*. Springer.

HUGOSON, P., AND NILSSON, A. 2007. *Hybrid Fluid Simulation*. Master's thesis, Hgskolan Kalmar.

IRVING, G., GUENDELMAN, E., LOSASSO, F., AND FEDKIW, R. 2006. Efficient simulation of large bodies of water by coupling two and three dimensional techniques. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, ACM, New York, NY, USA, 805–811.

KARAFYLLIDIS, I., AND THANAILAKIS, A. 1997. A model for predicting forest fire spreading using cellular automata. *Ecological Modelling 99*, 1, 87–97.

KASS, M., AND MILLER, G. 1990. Rapid, stable fluid dynamics for computer graphics. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 49–57.

KPPEN, T., AND NIVFORS, A. 2005. *Realtidsrendering av stora vattenytor*. Master's thesis, Hgskolan Kalmar.

LAYTON, A. T., AND VAN DE PANNE, M. 2002. A numerically efficient and stable algorithm for animating water waves. In *The Visual Computer*, Springer-Verlag, 41–53.

LEE, H., KIM, L., MEYER, M., AND DESBRUN, M. 2001. Meshes on fire. In *Proceedings of the Eurographic workshop on Computer animation and simulation*, Springer-Verlag New York, Inc., New York, NY, USA, 75–84.

LIU, P.-S., AND CHOU, Y.-H. 1997. A grid automation of wildfire growth simulation.

LOMAX, H., THOMAS H. PULLIAM, AND ZINGG, D. W. 2001. *Fundamentals of Computational Fluid Dynamics*. Springer-Verlag.

LUEBKE, D., REDDY, M., COHEN, J. D., VARSHNEY, A., WATSON, B., AND HUEBNER, R. 2003. *Level of Detail for 3D Graphics*. Morgan Kaufmann.

MAES, M. M., FUJIMOTO, T., AND CHIBA, N. 2006. Efficient animation of water flow on irregular terrains. In *GRAPHITE '06: Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*, ACM, New York, NY, USA, 107–115.

MITTRING, M. 2007. Finding next gen: Cryengine 2. In *SIGGRAPH '07: ACM SIGGRAPH 2007 courses*, ACM, New York, NY, USA, 97–121.

MLLER-FISCHER, M. 2007. Real time fluids in games. SIGGRAPH.

MÜLLER, M., SCHIRM, S., AND DUTHALER, S. 2007. Screen space meshes. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 9–15.

NOE, K. O., 2004. Implementing rapid, stable fluid dynamics on the gpu.

O'BRIEN, J. F., AND HODGINS, J. K. 1995. Dynamic simulation of splashing fluids. In *Computer Animation '95*, 198–205.

ROGERS, S. E., AND KWAK, D. 1991. An upwind differencing scheme for the incompressible navier-stokes equations. *Appl. Numer. Math. 8*, 1, 43–64.

STAM, J. 1999. Stable fluids. In *Siggraph 1999, Computer Graphics Proceedings*, Addison Wesley Longman, Los Angeles, A. Rockwood, 121–128.

STAM, J. 2001. A simple fluid solver based on the fft. *J. Graph. Tools 6*, 2, 43–52.

STAM, J., 2003. Real-time fluid dynamics for games.

TESSENDORF, J. 1999. Simulating ocean water. In *SIGGRAPH Course Notes*, SIGGRAPH.

THUREY, N., MÜLLER-FISCHER, M., SCHIRM, S., AND GROSS, M. 2007. Real-time breakingwaves for shallow water simulations. In *PG '07: Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, IEEE Computer Society, Washington, DC, USA, 39–46.

WEISSTEIN, E. W. Distribution function. accessed 12-March-2008.

WEISSTEIN, E. W. Probability function. accessed 12-March-2008.

WIKLUND, O. 2007. *A Computational Study of a Finite Element Method for Two Phase Flow*. Master's thesis, Kungliga Tekniska Hgskolan.

WONG, A. C. 2003. *The Moving Contact Line in a Shallow Water Simulation*. Master's thesis, University of British Columbia.

W.W., H., R.H., G., M.G., T., W.H., R., AND D.G., D. 2000. Simulating fire patterns in heterogeneous landscapes. *Ecological Modelling 135*, 2-3, 243–263.

Core-based morphing algorithm for triangle meshes

Martina Málková*
Ivana Kolingerová†
Jindřich Parus‡
Department of Computer Science and Engineering, University of West Bohemia

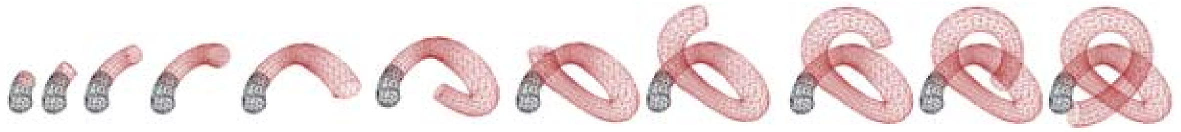


Figure 1: *Morphing of a knot - the knot (target mesh) grows out from the piece of rope (source mesh) in a natural way*

Abstract

This paper presents a method for the metamorphosis of genus-0 triangle meshes based on their intersection. It is an extension of our previous 2D algorithm [Málková, 2007]. Our algorithm is designed to simulate growing processes, therefore it is useful for morphing objects, where the user expects some parts of the latter object to grow out from the former one (e.g. a head with and without horns). The user can influence the algorithm's behavior by changing the mutual position of the objects, while the results are easily predictable.

CR Categories: I.3.5 [Computational Geometry and Object Modeling]: Boundary representations— [I.3.7]: Three-Dimensional Graphics and Realism—Animation

Keywords: morphing, triangle meshes, mesh intersection

1 Introduction

The goal of morphing is to compute a shape transformation (represented by an animation) between two shapes. There are usually many different ways how to deform one shape to another. The goal is to choose such a transformation which is visually plausible. Since morphing has a wide use in many types of applications (computer animation, design, special effects in movies, image compression and data visualisation), each of them requires different behaviour of the resulting morph and so defines the visual plausibility in a different way. Sometimes we might want to explode the initial shape into particles and form the final shape from the particles; sometimes we might want a continuous transformation during which the volume is preserved, etc. The algorithms usually concentrate on one type of application, trying to meet all its expectations. The most often expectation is to preserve the common features of similar shapes, so most algorithms put stress on this area. However, in some cases it is not possible to align all features (e.g., a body with and without a tail). In such cases, the user usually expects the nonaligned part to grow out of the rest, which is something that he/she can hardly obtain (or sometimes cannot obtain) from the current algorithms. To fill this area, we designed a new morphing algorithm, which has a "growing-like" nature, i.e., we focuse on a shape transformation that mimics growing (or disappearing) of some parts. Unlike most other algorithms, ours is not limited by star-shaped objects and can be used for all genus-0 objects.

After defining the input shapes, the computation of a morph consists of solving two distinct subproblems - establishing the correspondence of vertices and setting the vertex paths of the corresponding vertices, together with the dynamics of the transformation, describing how the vertices move in time. Although the properly computed vertex path can lead to establishing the expected transformation between the corresponding vertices, the crucial problem here is to automatically establish such correspondence that is similar to the one the user would enter manually.

Most algorithms let the user influence the result by adding some constraints or by manually defining correspondence of some concrete vertices. This process can be long and not intuitive, the user sometimes cannot easily predict which vertices he/she should define to achieve his expectations from the result. Our algorithm offers an intuitive tool for influencing the vertex correspondence - the user can achieve different results by changing the mutual position of the objects, while the result is always easily predictable.

The main steps of our algorithm are as follows: after the mutual position of the objects is defined, it computes the intersection of the objects, defining the *core* of the morphing. The core is a fixed part that is constant and will not change its shape during the morphing process. When transforming between the source and the target object, parts of the source object disappear in the core (i.e., they die away) whereas parts of the target object grow out of the core. The process of disappearing can be viewed as a reversed process of growing, thus algorithmically we need to solve only one process.

The paper describes an extension of our polygon morphing algorithm into 3D, where its input objects are triangle meshes. The meshes are assumed to be closed, however the presented algorithm is not limited only for closed meshes, if we are able to compute or manually define their intersection.

The paper is organized as follows. Section 2 describes related work in the area of mesh morphing. Section 3 describes the method it-

[*] mmalkov@kiv.zcu.cz

[†] kolinger@kiv.zcu.cz ; Work has been supported by the Ministry of Education, Youth and Sports of the Czech Republic under the research program LC-06008 (Centre for Computer Graphics)

[‡] jparus@kiv.zcu.cz

self and discusses two possible strategies of the process dynamics. The following Section 4 shows results of the algorithm and demonstrates typical use of the techniques described in Section 3. The last Section 5 describes conclusions and suggestions for future work.

2 Related Work

Warping and morphing have a long tradition in Computer Graphics and the full description is out of the scope of this paper. We refer reader to [Gomes et al., 1999] for a detailed description. Here we describe the work related to mesh morphing. The techniques described deal with morphing of triangular meshes, however, we can use those techniques for polygonal meshes as well if we triangulate them first.

As Alexa describes in his overview of mesh morphing algorithms [Alexa, 2002], most of the algorithms consist of three main steps:

1. **Establish a correspondence between the meshes.** Decide which vertex of the mesh M_0 corresponds to which one of the mesh M_1. This is usually the crucial step of the whole process.

2. **Generating a *supermesh*.** A supermesh is a mesh that represents both M_0 and M_1.

3. **Creating paths $V(t), t \in< 0, 1 >$ for the vertices.** Usually the algorithms use an interpolation of corresponding vertices, mostly a simple linear interpolation. However, as is discussed in [Gomes et al., 1999], such solution have problems when computing rotational morphing; when we morph between two line segments, both of them of the same length but one rotated, the line segment shorten during the morphing process, which is something we do not expect.

In [Kent and Carlson, 1992], the authors first project the objects onto a unit sphere. The projection used depends on the type of the input objects and is discussed for the most of the genus-0 objects(convex and star-shaped objects, objects of revolution and extruded objects). The correspondence is established by merging the topologies of the input objects. The merging process is done by clipping the projected faces of one model to the projected faces of the other. The paths for the corresponding vertices are created by either a linear interpolation, or using a Hermite spline with its tangent vectors equal to the vertex normals.

Alexa uses the idea of Kent et al. and presents another correspondence-based algorithm for morphing polyhedra [Alexa, 2000]. After triangulating the polyhedron, he examines the spherical projection. Because the method is designed for all genus-0 meshes, the projection may produce some overlapping edges (*foldovers*). To remove the foldovers, the relaxation process is introduced. The relaxation works iteratively, moving in each step each vertex to the center of its neighbors' positions in the previous step. Some vertices on the sphere need to be fixed to avoid the vertices converge to one position, but there is still some possibility that the relaxation will degenerate. The resulting embeddings are merged to get the supermesh. The supermesh is deformed to have the shape of the source (and equivalently target) mesh by setting its vertex positions. The vertices of the source mesh remain the same, and the positions of the vertices of the target mesh are computed by using the barycentric coordinates.

The algorithm presented in [Ahn et al., 2004] tries to enhance the correspondence-based morphing by decreasing the number of vertices of the supermesh. It uses the spherical embedding from the previous Alexa's approach (and therefore it is limited by the same class of objects) to find the correspondence between M_0 and M_1. After projecting M_0 and M_1 onto the unit sphere, M_0' (and similarly M_1') is constructed by mapping the vertices of M_1 onto the surface of M_0. Additional edge swapping of the created edges is done to reduce the difference between M_0' and M_1. Then, the sequence of connectivity transformations between M_0' and M_1' is computed by using an adaptation of the algorithm from [Hanke et al., 1996], resulting in a graph of edge swaps that needs to be done during the transformation. Each swap is realized by a *geomorph* [Hoppe, 1996] to make it smooth. The disadvantage of this approach is the need of computations during the creation of the in-between meshes, which slows down the resulting animation. On the other side, the resulting meshes contain much smaller number of vertices in comparison to other approaches using a fixed connectivity. The visual results are claimed to be similar to Alexa's approach.

A noncorrespondence based approach is described in [Cohen-Or et al., 1998]. The method needs the user to define corresponding control points on the input objects first. These points are used to define such warp function $\{W_t\}_{t=[0,1]}$ that $W_1(M_0)$ approximates M_1 as well as possible. The warp function consists of a rigid (rotation, translation) and elastic transformation of M_0. The warped object is rasterized into a binary discrete volumetric representation and converted into a 3D distance field by a method presented in [Levin, 1987]. Both M_0 and M_1 are represented as discrete distance field (DF) volumes, and the intermediate object (supermesh) is constructed by generating its DF-volume and extracting its surface. The quality of the resulting morph highly depends on a proper warp - if the corresponding points of the two objects are correctly aligned by the warp, it produces the expected results. Otherwise, it may produce results that are far away from the expected ones, sometimes containing parts that unexpectedly disappear and reappear. Also the creation of volumetric representation can consume a large storage space for meshes with a large number of triangles. The main benefit of this method is that it does not require the input objects to be of the same topological genus.

3 The Proposed Solution

3.1 General Idea

For better understanding of the following text, let us briefly recall the 2D algorithm, from which the 3D method was extended [Málková, 2007]. The input of the 2D algorithm are two simple polygons. The intersection of the input polygons is computed, resulting in a simple polygon called *core*. The core is the only part that does not change during the morphing process, the parts of the input polygons that are outside the core either grow out from the core or disappear in the core. Such parts are also simple polygons, whose boundary consists of a polygonal chain C_{in}, containing vertices and edges common to the part and the core, and a polygonal chain C_{out} containing vertices and edges lying outside the core. The chains C_{in} and C_{out} meet at exactly two vertices, called intersection vertices. The goal is to compute the morphing sequence between C_{in} and C_{out}. For each vertex of C_{out}, so called *vertex path* is computed. The vertex path consists of positions, defining where the vertex travels during the morphing process. If the vertex path contains only two positions (the initial position and the final position of the vertex), it represents *vertex-to-vertex correspondence*. But, for example, to represent the growth of non-convex shapes without self-intersections, the vertex path needs to contain more positions. The vertex paths define both the vertices correspondences (the first and the last positions in the path are the ones of the corresponding vertices) and their key positions during the animation. To view the final morph, we only need to interpolate between the positions in vertex paths according to the given time.

Brief description of our 3D extension is quite similar. As the input, we have two objects A, B defined by their triangle mesh boundaries $\delta A, \delta B$. The intersection of both input objects (*core* C) is computed. The boundary of the intersection is also a triangle mesh δC. The parts of the objects that are outside the core are computed as the differences $P = A - C, Q = B - C$. Each part's boundary consists of a mesh C_{in} containing vertices and triangles common to the part and the core, and a mesh C_{out} containing vertices and triangles lying outside the core. Those two meshes meet at $1..n$ closed polygonal chains, called *intersection chains*. The goal is to compute the morphing sequence between C_{out} and C_{in}. Again, the vertex path is computed for each vertex of C_{out} and the final morph at a given time is computed by interpolating between the coordinates in the vertex path of each vertex.

Now, let us describe the algorithm in detail. As the input, we have two objects A, B. The core is computed as $C = A \cap B$. It can consist of $C = (0, \ldots, c-1)$ disjoint parts. Because our algorithm is dependent on the existence of core, it cannot solve the case when $C = \emptyset$. It is designed for the case when $c = 1$. However, we can solve the problem of multiple parts by choosing one representative part as a core. Therefore, let us suppose in the following text that the core consists of only one part.

When the core C is computed, we compute the difference $P = A - B$, which will disappear in the core, and the difference $Q = B - A$, which will grow out of the core. In the following description, we will concentrate only on solving the disappearing of one part P_i, $P = \bigcup P_i$. The other parts from P are computed equivalently, as well as the parts Q, $Q = \bigcup Q_j$. We achieve the growing of part Q_i by simply reversing the time plan.

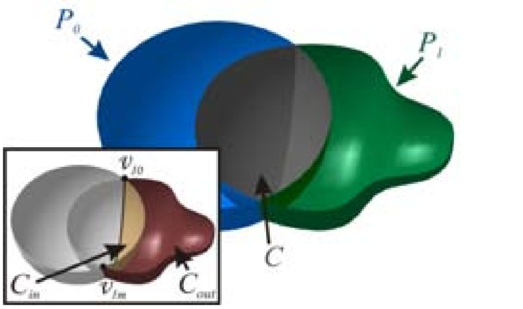
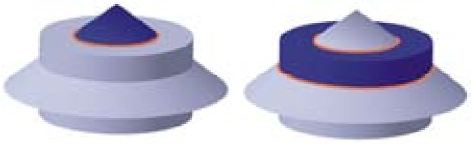


Figure 2: *General terms: Core C, part $P_i = C_{out} \cup C_{in}$, intersection vertices v_{Ii}.*

δP_i consists of two surfaces C_{in}, C_{out} (shown in Figure 2). The surfaces C_{in}, C_{out} are separated by closed polygonal chains, let us call them *intersection chains* I. There can be an arbitrary number of intersection chains $I = \bigcup I_i, i = 0 \ldots n$. Figure 3 shows an example of a part with one intersection chain (Figure 3a) and two intersection chains (Figure 3b), which are the most often cases.

By morphing the surface C_{out} to the surface C_{in} we achieve the effect of disappearing of the part P_i in the core C.

Any method can be used for morphing C_{out} to C_{in}, ours uses a concept of so-called *topological distance* of a vertex. If we define a distance $d(v_i, v_j)$ between vertices v_i and v_j as the minimal number of edges on the mesh between v_i and v_j, we can compute the topological distance of a vertex v_i as its minimal distance $d_{min} = min(d(v_i, v_j)), v_j \in I_k), k = 1 \ldots n$, where n is the number of intersection chains. The topological distance therefore represents the length of the shortest path from the vertex to any intersection chain.



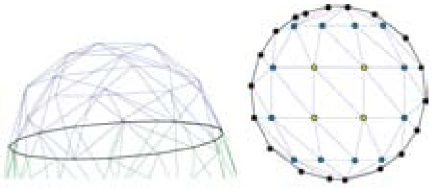
(a) one intersection chain (b) two intersection chains

Figure 3: *There can be an arbitrary number of the part's intersection chains (orange).*

The topological distances are computed by the *Breadth-first search* algorithm, where the search begins at the intersection chain(s). The vertices of the intersection chain(s) are assigned a topological distance of zero, and put into the stack. Than one by one, the vertices are popped from the stack, and for each vertex v_i (with a topological distance t_i), we go through its neighbors. If the neighbor has greater topological distance than $v_i + 1$, we assign it the new distance and put it in the stack. The algorithm is shown in Figure 4. An example of computed topological distances can be seen in Figure 5.

```
Input: List of vertices of the intersection chains of the current part
    $I = \bigcup v_{Ii}$, list of vertices $C_i = (v_1, \ldots, v_N)$, for which the
    topological distance should be computed. Stack $S = \emptyset$.
Output: List of vertices $C_i = (v_1, \ldots, v_N)$ with assigned topological distances $(d_1, \ldots, d_N)$.
The algorithm:
    1. Initialization: For all $v_{Ii} \in I$: $d_{Ii} = 0$, push $v_{Ii}$ in $S$.
       For all $v_j \in C_i$: $d_j = \infty$.
    2. Pop a vertex $v_i$ from the top of the stack. Go through
       all its neighbors, for each neighbor $v_j$: If $d_j > d_i + 1$,
       $d_j = d_i + 1$ and push $v_j$ in $S$.
    3. If $S \neq \emptyset$, continue by 2. Otherwise the algorithm is
       finished.
```

Figure 4: The algorithm for computing topological distances.



(a) (b)

Figure 5: *An example of topological distances in 3D (a) side view of the part (blue), the intersection chain (black) (b) top view - computed topological distances: - intersection points ($d_i = 0$ black, $d_i = 1$ blue, $d_i = 2$ yellow.*

After the topological distances are computed for both C_{in} and C_{out}, we compute the vertex paths of the vertices of C_{out}. These vertex paths can be a simple one-to-one correspondence between the vertices of C_{out} and C_{in}, in which case they contain only the positions of the corresponding vertices, but they can also be more complex. For the vertex path computation, we extended our 2D method called *Perimeter growing*, which searches the positions of the vertices among the existing vertices of C_{out}. This method will be described in Section 3.2 in detail.

After the computation of vertex paths, we need to merge the parts and the core into the resulting supermesh. The reason why we cannot just take the whole parts and the core and produce the final morphing sequence is that during the animation, the vertices of C_{in} rest at their positions during the time, while the vertices of C_{out} change their positions (travel towards their corresponding vertices in C_{in}). While moving, some vertices of C_{out} can cross an edge of C_{in} - and at that time, the vertices and edges that were inside and should not be visible, are now visible (see Figure 6).

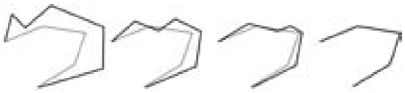


Figure 6: *Not removing C_{in} may result in unwanted self-intersections: C_{in} (grey), C_{out} (black) (in 2D for simplicity)*

When the result is merged, the animation can run by using the time plan. The time plan computes the actual velocity of each vertex depending on a time and the chosen dynamics of the animation. The different dynamics setting and their behavior will be discussed in Section 3.3.

3.2 Perimeter Growing

As the vertex paths are represented as a list of desired positions of the vertex during the time, the goal of the Perimeter Growing is to compute such a list for each vertex of a given part that the vertex paths contain only the positions of existing vertices of the part. Therefore during the final animation, all the vertices follow the boundary of the part. To try to avoid self-intersections, the vertices should take the shortest possible way they can to reach their corresponding vertices from C_{in}. Such an approach should lead to a morphing sequence, where C_{out} continuously grows out of C_{in}, following the perimeter of the original part.

To find a vertex path of a vertex v_i, we first need to find out where should the path lead - decide about the corresponding vertex. In 2D, the correspondences were solved from the intersection vertices, depending on the topological distance of each vertex and the count of the vertices of C_{in} and C_out. In a special case when C_{in} and C_{out} had the same number of vertices (Figure 7a), a vertex with d_i from C_{in} corresponded to a vertex with d_i from C_{out}. In other cases, either more vertices from C_{out} corresponded to one vertex from C_{in} (Figure 7c, or some vertices from C_{out} were duplicated to cover all the vertices from C_{in} (Figure 7b). The vertex paths were set up only for vertices from C_{out}, each of them containing the coordinates of the corresponding vertex from C_{in} as its last element.

As each vertex of the part in 2D had only two neighbors, only two vertices had the same topological distance and so knowing the topological distance was enough to be able to solve the correspondence. However, in 3D, the solution is not so clear, because more than two vertices can have the same topological distance. Let us discuss the following example: In Figure 8, we have four vertices with d_{max} (A, B, C, D) and three vertices with d_{min} (E, F, G), and we need to solve the correspondence problem within them. Visually, we can decide that A should end at E; B, C at F and D at G. But, unfortunately, we do not have any other information about the vertices than their topological distance, and so we are unable to decide this automatically. We could try to decide it according to the neighbors, but we still would not know whether A should end at E or G. Or worse, also E and G could be neighbors and then neighboring information would not help much.

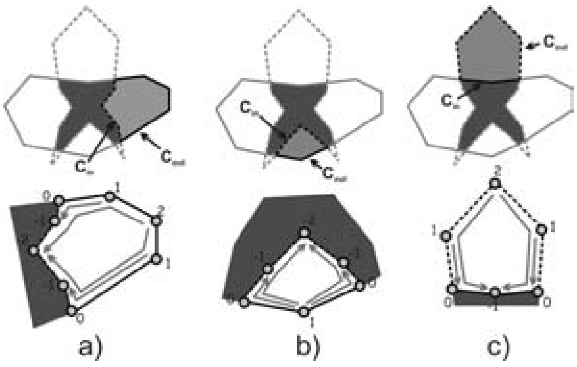


Figure 7: *Solving correspondences in 2D when a) C_{out} and C_{in} have the same number of vertices, b) C_{in} has more vertices than C_{out}, c) C_{out} contains more vertices than C_{in} (dark grey: core, grey: processed part, arrows are connecting the corresponding vertices)*

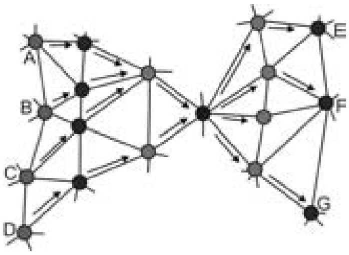


Figure 8: *Correspondence problem at one intersection vertex (black, other vertices are colored according to their topological distance: ±1 light blue, ±2 blue, 3 green).*

However, we are able to solve the correspondence problem to the intersection vertices while computing the topological distances. When a vertex v_i assigns to its neighbor v_j some topological distance, v_j is assigned the vertex path containing the vertex path of v_i extended by the coordinates of v_i (Figure 11, step 2).

This way we are able to compute the transformation from C_{out} to the intersection line. In the same manner, we can compute the vertex paths for C_{in} and therefore its transformation to the intersection line as well.

So although we are not able to solve the correspondence between C_{in} and C_{out} directly, we can piece the resulting morphing sequence from three parts (see Figure 9): In the first part, $(0, t - \epsilon)$, vertices of C_{out} move towards the intersection vertices. At $t - \epsilon$, the resulting morph appears to contain only the intersection vertices (but there are also the vertices of C_{out}, which are at the same positions as the intersection vertices), connected by edges defined by C_{out}. Let us call it S_0. In the last part, $(t + \epsilon, 0)$, vertices of C_{in} move from the intersection line to their original positions (their vertex paths are reversed). At $t + \epsilon$, the resulting morph appears to have only the intersection vertices, connected by edges defined by the C_{in}. Let us call it S_1. During the time $(t - \epsilon, t + \epsilon)$, we need to morph from S_0 to S_1, which are two triangular meshes with the same vertices, but a different connectivity. 2ϵ is the time amount spent on the connectivity change, its value depends on how fast we want the change to run over.

The last part to solve is to transform the connectivity of S_0 to S_1 by an edge swap sequence, as is presented in [Ahn et al., 2004].

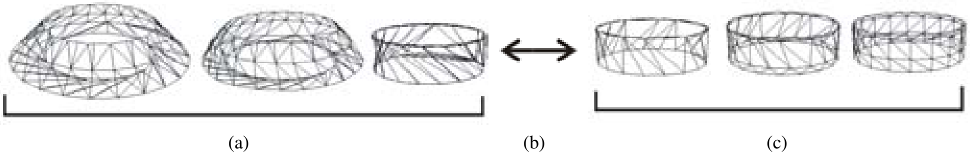


Figure 9: *The resulting morphing sequence consists of three parts: (a) the outside part morphs towards the intersection vertices, (b) connectivity change, (c) the inside part morphs from the intersection vertices.*

The continuous swap is created by applying geomorphs [Hoppe, 1996] (Figure 10). As geomorphs can be used only to swap one edge, the dependency graph of edge swaps is constructed to allow the transformation to proceed in a minimal possible time.

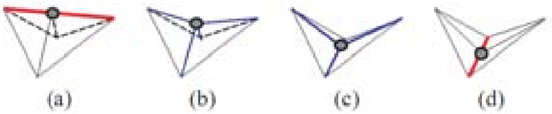


Figure 10: *Using geomorph to swap the edges: (a) create the vertex at the center of the edge to be swapped, (b)&(c) the vertex moves during the geomorph, (d) the edge has been swapped so the vertex can be removed*

(From [Ahn et al., 2004])

The pseudocode of the perimeter algorithm is shown in Figure 11. Note that the algorithm has only small changes from the one computing the topological distances (Figure 5): the assignments of the vertex path in steps 1,2.

```
Input: Part $P_i$ in the form of three lists: List of vertices of the intersection chains $I = \bigcup v_{Ii}$, list of vertices on the core $C_{in}$ and list of vertices outside the core $C_{out}$. Stack $S = \emptyset$.
Output: Part $P_i$, where each vertex of $C_{out}$ and $C_{in}$ has its vertex path computed in a form of a list of coordinates $pv_i = v_0, \ldots, v_{n-1}$
The algorithm:
    1. Initialization: For all $v_{Ii} \in I$: $d_{Ii} = 0$, $pv_i = v_{Ii}$, push $v_{Ii}$ in $S$. For all $v_j \in C_{out}, C_{in}$: $d_j = \infty$.
    2. Pop a vertex $v_i$ from the top of the stack. Go through all its neighbors, for each neighbor $v_j$: If $d_j > d_i + 1$, $d_j = d_i + 1$, $vp_j = v_j \bigcup vp_i$ and push $v_j$ in $S$.
    3. If $S \neq \emptyset$, continue by 2. Otherwise the algorithm is finished.
```

Figure 11:The Perimeter algorithm.

The main advantage of the algorithm is its simplicity and usage for arbitrary types of parts. The algorithm produces best results when the mesh is uniform, which leads into its often use together with some remeshing algorithm.

3.3 Time Plan

After computing the vertex paths and creating the supermesh, the task is to compute the positions of vertices according to the current time (create the final animation or the morphed shape). The way the vertex paths are handled is called the time plan. The time plan decides when each vertex travels and how it travels. The former offers at least two possibilities - first, the vertex moves all the time and if it has n positions in its path, it has $(n-1)/t$ time to travel between each pair of positions, where t means the whole amount of time. Second possibility is to assign to each vertex a different amount of time, according to the number of positions it has in its vertex path. The vertex with the longest vertex path travels all the time, and the other vertices move only during their portion of the time.

Along the time distribution, the time plan decides about the trajectory of the vertex between the key positions defined by the vertex paths. The actual position of the vertex between two positions from its vertex path can be computed either by a linear interpolation, or by any other interpolating method.

The choice of the concrete time plan depends on the output of the method for the vertex path computation and on our expectations of its final behavior. For the Perimeter growing, we decided to use the traditional linear interpolation along with assigning each vertex different amount of time depending on the length of its vertex path.

As was already told, the Perimeter growing produces two supermeshes, one for the time $(0, \frac{t}{2} - \delta)$ and the other for $(\frac{t}{2} + \delta)$. Let us denote the amount of time for one supermesh t_s. Each vertex v_i in the supermesh has $d_i + 1$ coordinates in its vertex path (where d_i is its topological distance). The vertex with the longest vertex path is the one with d_{max}. The maximal topological distance for each part P_i can be found during the distances computation. The amount of time for each vertex is then computed as $t_i = (0, \frac{t_s}{d_{max}})$. The resulting time sequence is shown in Figure 12 (top view of the part from 5 is used). First, all the vertices with $d_i > 0$ travel to the positions of the ones with $d_i = 1$, where such vertices stop and the others continue to the positions of the vertices with $d_i = 2$ and so on.

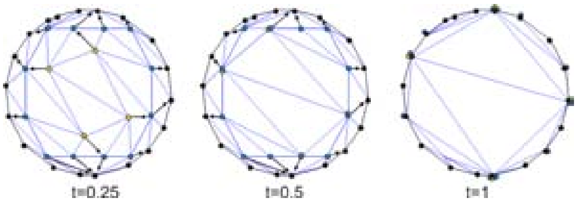


Figure 12: *The morphing sequence for one part, where we used the time plan with linear interpolation and different amounts of time for each vertex (the vertex paths are computed by the Perimeter growing).*

4 Results

In this section, we will discuss the cases where the use of our algorithm brings benefits. As the output of our algorithm depends on the mutual position of the meshes and therefore on the shapes

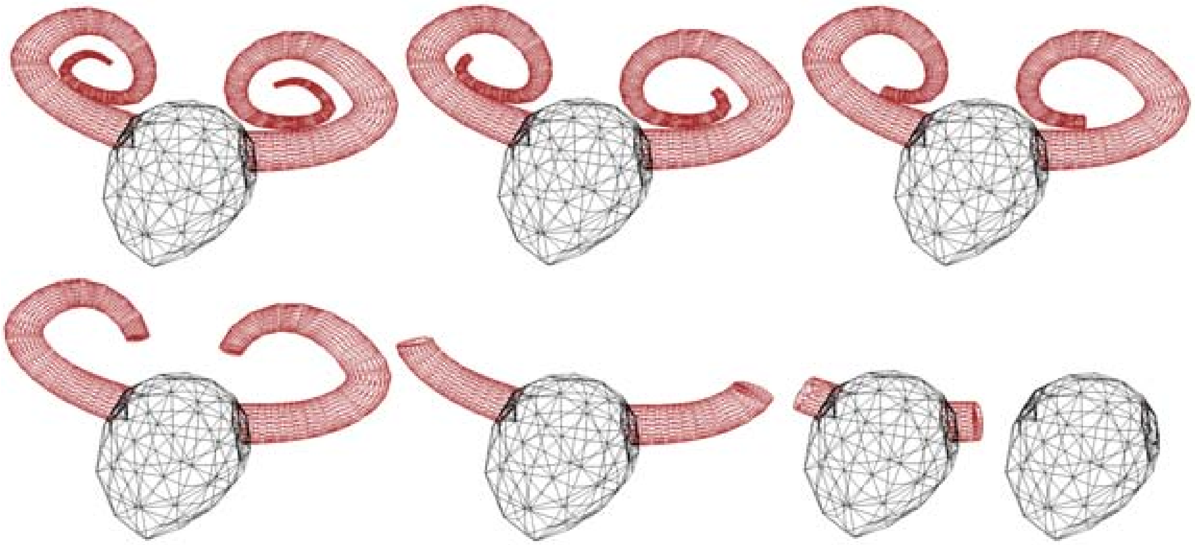


Figure 13: *Two horns (parts, red) morph from the head (core, black).*

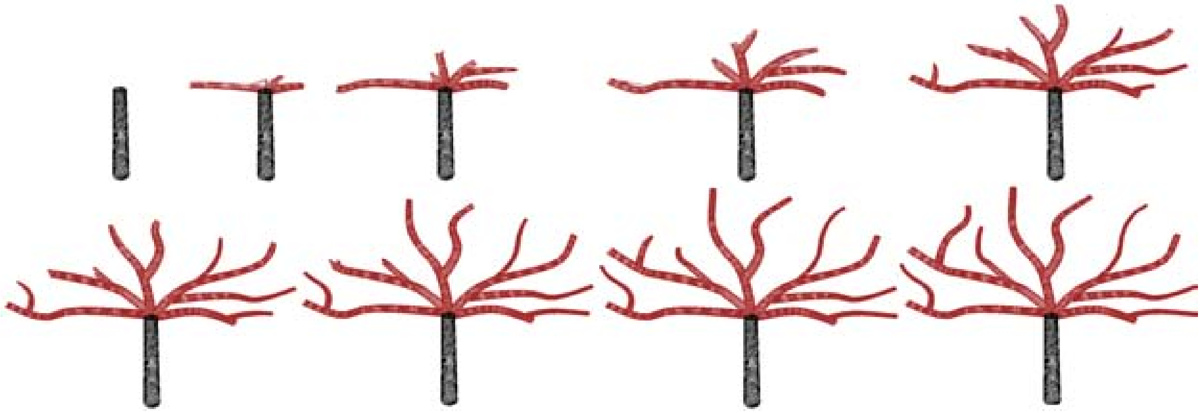


Figure 14: *An example of morphing a complexly branched part (red)*

of the parts P, Q, we will discuss its behaviour according to the shapes of the parts and not the types of meshes. The presented method (Perimeter growing) is used independently for each part, so we will show its behavior on one part's shape at a time.

Let us recall, that we should not use our algorithm, when we do not want the output to grow out from the mesh intersection, but we suppose the algorithm to align the corresponding features and morph between them (i.e. morph between faces with different expressions or bent and straight finger). Our algorithm is designed only for creating the growing behavior of the part. In special cases, the growing behavior can be similar to the feature one, but in general it is a completely different output.

The presented Perimeter growing method produces better results for thin and long parts (independently on the number of branches and curves of the part), because it ”cuts” the top of the part during growing. Figures 1, 13 show examples of morphing a long curved part, consisting of only one branch. Also more branches are not a problem, as Figure 14 shows, where the algorithm produces only small artefacts around the branch joinings.

In all the examples, the core C was similar to one of the meshes, which suggests the idea of the use of our algorithm - to add something else to an existing shape and make it grow from it, or similarily let something of an existing shape disappear in the rest.

Animation videos of the examples can be seen at http://herakles.zcu.cz/research/morphing.

In the presented examples, we followed two current limitations of the algorithm. The former is, that only two input meshes are considered. The latter has been already discussed in Section 3 - the intersection of the input meshes should consist of only one part, if it does not, the algorithm chooses one of the parts and consider it the only one (Figure 15a,b). However, this solution always leads to self-intersections in the area of the other parts. To avoid the self-intersections, we would have to split such parts of the objects that connect two (or more) parts of the core. This would lead to a proper result without self-intersections, however, the first object would first split into several objects, which would then merge to form the target object. We did not want to achieve this type of effect, however it would be possible future work in case of its practical need.

The algorithm itself is able to deal with more than two objects, but there are some limitations. It handles the special case, when the objects have the same intersection (Figure 15c). We only need to define which objects are to disappear in the core and which ones are to grow out from it. When there are distinct pairs of objects and each pair has an intersection consisting of only one part (Figure 15d), the algorithm can be used iteratively for each pair. However, if the input objects intersect at several places while not having any part of the intersection common for all of them (Figure 15e), the algorithm will not work.

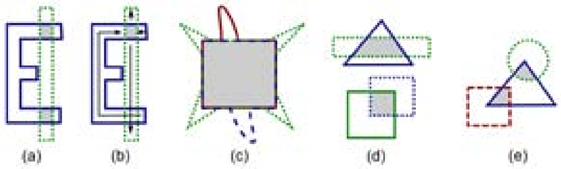


Figure 15: *Limitations of the algorithm (shown in 2D for simplicity): (a,b) more parts of the core (grey) lead to choosing only one, (c)more input objects with the same core, (d) distinct pairs of objects which can be solved separately, (e) several cores without a common one*

Although the algorithm itself is not much usable for more input objects, its nature offers an extension, where one object is given as the core, and the other objects are considered to be the parts that either grow out or disappear from the core. Each object is required to have an intersection with the core object, and this intersection is clipped out. In such a way, the algorithm can have an infinite number of input objects. The idea is sketched in Figure 16 (the core object is filled with grey, other objects are only outlined). This extension has not been implemented yet and belongs to our future work. By using this different approach to define the input meshes, the user would be able to first model the core (i.e., a body) and then add the parts separately (i.e., horns, a tail), placing them at a desired position.

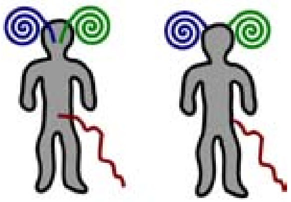


Figure 16: *A different approach to defining input meshes, where the user explicitly defines the core (filled with grey) and the parts (in 2D for simplicity): (a) input objects (b) clipped input objects representing the parts*

5 Conclusion and Future Work

We have presented a novel approach to morphing meshes. The goal of our approach was to offer a tool for animating growing processes, which is an area of morphing that is not explored. Our algorithm solves the problem of morphing between two meshes in an unusual way. It does not compute the correspondence between the whole two objects, but it first computes an intersection of the objects, subtracts the intersection from the original objects and so it decomposes them into several parts lying outside the intersection. Then it computes the correspondence separately for each part - correspondence between the vertices lying outside the intersection and the vertices lying on the intersection. This way, the morphing is established as a process of disappearing of the parts of the latter object and growing of the parts of the former object.

We presented one method, called Perimeter growing, for computing the transformation of each part. This method computes the correspondences of vertices and the vertex paths at the same time. It is based on the topological distances of the vertices. As the topological distances do not cover the euclidean distance information, the input meshes should be uniform to obtain best results. The method has strong advantages when we use it for long curved parts, but it is usable also for other shapes of the parts.

The experiments confirmed that our algorithm (in combination with the Perimeter growing method) is suitable for the cases, where the user expects some parts of the object to grow out from the intersection or disappear in it. It can be used also for some other than grow-like cases, but it is not useful for the objects that are of the same shape and they are only transformed (rotated, translated etc.).

There are still some implementation parts to finish. To be able to use the Perimeter algorithm, we need to implement the Ahn's algorithm [Ahn et al., 2004] to continuously change the connectivity between the two in-between meshes in Perimeter growing. The same algorithm should be used to handle the connectivity in the Projection growing, so that its result for the pieces lying on the core are precise, not only approximations as it is now. And last, the merging algorithm should be established for the 3D parts.

The intersection of the two input objects has been computed manually in 3ds max 7, so part of the future work will be to implement this computation to be able to change the positions of the input objects in our program.

References

Ahn, M., Lee, S., and Seidel, H. (2004). Connectivity transformation for mesh metamorphosis. In *SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 75–82, New York, NY, USA. ACM.

Alexa, M. (2000). Merging polyhedral shapes with scattered features. *The Visual Computer*, 16(1):26–37.

Alexa, M. (2002). Recent advances in mesh morphing. *Computer Graphics Forum*, 21(2):173–197.

Cohen-Or, D., Solomovic, A., and Levin, D. (1998). Three-dimensional distance field metamorphosis. *ACM Transactions on Graphics*, 17:116–141.

Gomes, J., Darsa, L., Costa, B., and Velho, L. (1999). *Warping and morphing of graphical objects*. Morgan Kaufmann Publishers, Inc.

Hanke, S., Ottmann, T., and Schuierer, S. (1996). The edge-flipping distance of triangulations. *j-jucs*, 2(8):570–579.

Hoppe, H. (1996). Progressive meshes. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 99–108, New York, NY, USA. ACM.

Kent, J. R. and Carlson, W. E.and Parent, R. E. (1992). Shape transformation for polyhedral objects. *Computer Graphics*, 26:47–54.

Levin, D. (1987). Multidimensional reconstruction by set-valued approximation. *Clarendon Press Institute Of Mathematics And Its Applications Conference Series, Algorithms for approximation*, pages 421–431.

Málková, M. (2007). A new core-based morphing algorithm for polygons. Proceedings of CESCG.

Adaptive resolution in physics based virtual environments

M. Servin[*] Umeå University

C. Lacoursière[†] Umeå University

F. Nordfelth[‡] Algoryx Simulation

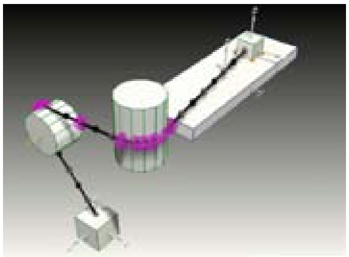


Figure 1: *Example from application including a wire with adaptive resolution. The system has two boxes connected by a wire routed over two cylindrical wheels that are free to rotate. Black dots shows lumped wire masses and magenta dots show contact nodes. Numerical robustness is secured by recognizing unstable states and simplifying the system to lower wire resolution for which the system is stable.*

Abstract

We propose a systematic approach to adaptive resolution in physics based virtual environments (VEs) that combines the conventional requirements of realtime performance, visual appearance with important requirements on the physical simulation, such as accuracy and numerical robustness. In particular, we argue that adaptive resolution is a key element to achieve robustness in fixed time-step VEs. The idea is to adaptively substitute unstable subsystems with more simplified and robust models. The method is demonstrated on systems including stiff wires. The algorithm brings stability, realtime performance and preservation of the important physical invariants to the system. The application to general systems is discussed.

Keywords: adaptive resolution, virtual environment, physics based animation, fixed time-step, numerical stability

1 Introduction

In physics based virtual environments (VEs), the state space of possible configurations is necessarily large and the trajectories depend critically on both user interaction and the dynamics of simulated objects. This is particularly true for training system where the state space cannot be restricted, i.e., catastrophic situations resulting from operating errors must be reachable in the simulation to avoid false training. This makes it difficult to construct VEs that are guaranteed to be robust, numerically stable over the entire state space, run in realtime, and offer optimal visual appearance and accuracy for the purposed use. The problem increases in complexity when taking account of different hardware set-ups with different performance, and view-dependent level-of-detail.

We propose a systematic approach to adaptive resolution in physics based VEs that combines the conventional requirements of realtime performance, visual appearance with important requirements on the physical simulation, such as accuracy and numerical robustness.

1.1 Related work

There is a vast litterature on *adaptive resolution* techniques, or *level-of-detail* (LOD) algorithms, in the context of 3D computer graphics and virtual environment since the early work by [Clark 1976]. The difficulties associated with applying LOD and culling algorithms in *physics based* virtual environments has attracted little scientific attention, but was adressed at least partially by [Carlson and Hodgins 1997] and [Chenney and Forsyth 1997]. Previous approaches have focused on either adapting the *spatial* resolution for the sake of visual quality or accuracy, e.g., [Spillmann and Teschner 2008] and [Redon et al. 2005], or on adapting the *time step-size* to avoid numerical instabilities in stiff systems [Debunne et al. 2001]. The latter approach conflicts with the realtime performance requirement and fixed time-step designs common in VEs. The former technique based on spatial refinement can introduce instabilities. We have found no example in litterature that combines the requirements of realtime performance, visual appearance with the important requirements on the physical simulation, such as accuracy and numerical robustness.

1.2 Our contribution

Our approach uses the fact that *the tendency for instabilities decreases as a system is made more coarse-grained.* For many systems these instability thresholds can be predicted in advance and incorporated in an adaptive resolution scheme. In particular, we apply this idea successfully in interactive simulations including inextensible but otherwise flexible wires—one of many stiff systems that plague VEs with numerical instabilities [Servin and Lacoursière 2008]. We present an algorithm for finding the optimal spatial resolution for the discretized wire objects at each time-step. The optimality is based on a line quality measure that includes weighted requirements for numerical stability, limited computational time and system granularity (linked to both visual appearance and solution accuracy). The algorithm brings stability, realtime performance and preservation of the important physical invariants to the system. Available computational resources can also be used to increase the resolution of specific parts of the VE, as prioritized by the user. Adaptive resolution has also proved to be a key element in finding robust contact methods for wires. We discuss how this works and show application examples of the method in the context of a maritime training simulator. Finally, we discuss the use of this method

[*] e-mail: martin.servin@physics.umu.se

[†] e-mail:claude@hpc2n.umu.se

[‡] e-mail:fredrik@algoryx.se

to more general systems than wires.

2 Adaptive resolution

The basis of physics based VEs is now outlined and a general algorithm for adaptive resolution is constructed.

2.1 Multibody system dynamics

Physics based VEs are examples of multibody system simulations (MBSS) which computes the motion of the system at discrete timesteps from a set of equations of motion. The computed motion depends on the initial conditions, the external and internal forces and various types of constraints, e.g., for modeling of joints, motors and dry frictional contacts. The computed motion also includes a numerical error that depends on the time step size and on the numerical solver being used. For practical reasons many VEs are run with fixed time-step, say, $h = 1/60\ s$. Dynamics on time scales shorter than h cannot be perceived visually by the user and is considered to be unimportant. Computational time should thus not be spent on resolving the high frequency dynamics unless it is necessary for the overall behavior. Using variable time-step or different time-steps for the different subsystems is possible but generally not practical in the realtime context, and thus finds little support in existing code libraries for rendering, collisions detection, or physics engines. Only fixed-step simulation is considered in what follows.

We use the descriptor form of multibody system dynamics. The variables for the system are generalized position q, velocity $\dot{q}$ and Lagrange multiplier λ for the kinematic constraint vector $g(q, \dot{q}, t) = 0$. For simplicity we assume the multibody system to be a particle system. The extension of the theoretical framework to include also rigid and deformable solid bodies is fairly straight forward. Given that the system has N_p particles and N_c scalar constraints the system variables have dimension $\dim(q) = \dim(\dot{q}) = 3 \times N_p$ and $\dim(\lambda) = \dim(g) = N_c$.

The system Lagrangian is

$$\mathcal{L}(q, \dot{q}, \lambda, t) = \tfrac{1}{2}\dot{q}^T M \dot{q} - V(q) - \lambda^T g \qquad (1)$$

where M is the system mass matrix of dimension $3N_p \times 3N_p$, which is symmetric and positive definite and V is the potential energy of the system. The Euler-Lagrange equations derived from the least action principle are then

$$M\ddot{q} = -\nabla_q V + G^T \lambda \qquad (2)$$
$$g(q, \dot{q}, t) = 0, \qquad (3)$$

where $G \equiv \nabla_q g$ is the constraint Jacobian. We notate the system state at time t by $x(t) = (q, \dot{q}, \lambda)$.

The equations of motion (2) can be integrated using a variety of techniques. In particular, there is considerable choice for constraint satisfaction and stabilization, ranging from direct or iterative linear algebra techniques, penalty formulations, and constraint projection strategies. Our choice is the combination of a discrete-time variational technique described in [Kharevych et al. 2006], and constraint stabilization and regularization [Lacoursière 2007]. This yields a computational procedure to produce $x(t + h)$ given $x(t)$. The discrete-time variational integrators are derived from the least action principle rather than from discretization of the equations of motion. These steppers preserve many of the important physical invariants of the system by construction. This makes them generally more robust. At fixed time-step for instance, linear and angular momentum are preserved to machine precision, globally over the entire simulation. Examples of variational integrators are the Verlet stepper and symplectic Euler.

2.2 A general algorithm for adaptive resolution

Besides the time evolution of positions and velocities we now also consider the evolution of a variable number of bodies in the MBSS. We assume a resolution variable r that represents the state of resolution for the systems. This can be an integer, integer vector or more complicated set of integers depending on how each subsystem may be described at different levels. For each r the system has a specific number of particles and connectivity. We further assume a set of quality measures $Q(r)_\alpha > 0$, $\alpha = 1, 2, \ldots$, that each measures a specific quality of a specific subsystem. Good quality corresponds to values close to unity and poor quality corresponds to values close to zero. Next we list a number of qualities that can be included. *i) User perceived quality*, $Q_{\text{user}}(r, \texttt{view}, \texttt{inter})$. The subsystem which the user is focusing on should have high geometric level of detail and high functionality. User focus is determined through the camera view (`view`) and level of interaction with the subsystem (`inter`). *ii) Accuracy*, $Q_{\text{acc}}(r, x)$. This measures the accuracy of the numerical solution for a subsystem. Flexible systems, such as deformable solids or fluids, may require sub-division to maintain a specified accuracy when deformed. *iii) Computational time*, $Q_{\text{time}}(r, t_{\text{est}} t_{\text{lim}})$. During each time-step there is a fraction $t_{\text{lim}} < h$ of the step-size that may be spent on computing the dynamics. The actual time for computation for each subsystem for a given resolution r can be estimated to some number t_{est}. High quality is when $\sum t_{\text{est}} < t_{\text{lim}}$. *iv) Robustness*, $Q_{\text{rob}}(r, x, M, f)$. The risk of numerical instabilities—diverging or erratic changes in velocities or positions—can be estimated from the state x of the subsystem, possibly in combination with a model for the numerical stability of the subsystem at various levels of resolution. These models may typically also include system mass M and the forces in the system f. In section 3 we give examples of robustness quality measures for a specific class of system and we elaborate on the problem of estimating the numerical robustness in general systems further in section 4. As a cost function for keeping the application quality high we introduce $\frac{1}{2}\sum_\alpha w_\alpha Q(r)_\alpha^{-2}$ with weight coefficients $w_\alpha > 0$ for each quality.

The problem of evolving a VE with optimal quality can be treated as a problem of evolving the extended system with variables (x, r) and Lagrangian

$$\tilde{\mathcal{L}}(x, r, t) = \tilde{\mathcal{L}}(x, t) - \tfrac{1}{2}\sum_\alpha w_\alpha Q(r)_\alpha^{-2} \qquad (4)$$

The resolution variable r can be treated either as a continuous variable and rounded to integer values or as an integer variable. The extended time stepping algorithm involves solving a complicated nonlinear equation, e.g., using Newton-iterations and taking into account the dependence in $Q(r)$ on $(q, \dot{q})$ and any implicit dependency in $(q, \dot{q})$ on r. This self-consistent formulation has the advantage that it has a variational formulation and may thus be approached with variational integrators like the rest of the system, i.e., treat the resolution parameters as any other dynamic variable and thus preserve the physical invariants of the system even at the events of change in resolution. It may however be too time consuming to perform Newton-iterations and solve these equations exactly. Instead, as a proof of concept, we assume a weak coupling between the variables $x = (q, \dot{q}, \lambda)$ and r and solve for them separately. An algorithm for MBSS including adaptive resolution for optimal quality in this form is given in Algorithm 1. Our approximation is to use the locally optimal solution for r', i.e., $r' = \arg\min_r(\frac{1}{2}\sum_\alpha w_\alpha Q(r)_\alpha^{-2})$. It is critical that the system reconfiguration is constructed to preserve the important physical invariants of the system, most importantly the total momentum. Step 9 in the algorithm involves also reconfiguration of the system connectivity, e.g., given by the constraints $g(x)$.

Algorithm 1 Adapt resolution for optimal quality

```
system initialization $x$ and $r$
while VE running do
    user interaction $\rightarrow$ (view, inter)
    accumulate explicit forces $-\nabla_q V(x)$
    update contact data and constraint data $g(x)$
    step particles $x_{i-1} \rightarrow x_i$
    compute the quality function
    $Q(r) = (Q_{\text{user}}, Q_{\text{acc}}, Q_{\text{time}}, Q_{\text{rob}})$
    compute optimal system resolution $r'$
    reconfigure the system $(q_i, \dot{q}_i, r) \rightarrow (q_i', \dot{q}_i', r')$
end while
```

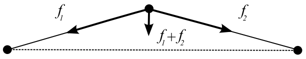


Figure 2: *The transversal force, $f_1 + f_2$, on the particles increases with the wire tension. Numerical instabilities develops when the wire tension and thus the oscillation frequencies become larger than the frequency of the numerical integration.*

3 Application to wires

In this section we describe physics based VEs containing wire systems with adaptive resolution for optimal quality. We model a wire by a set of constraints $0 = g^{\text{wire}} = (g_1^{\text{wire}} g_2^{\text{wire}}, \ldots, g_{N-1}^{\text{wire}})$ on a collection of $N \leq N_p$ particles, such that the particles are connected in line topology with pairwise distance constraints $0 = g_i^{wire} = |q_a - q_b| - l_i$ for maintained segment length i. The extension of this model to include stretch and bend elasticity and contact nodes is delayed to Sec. 3.3

Inextensible wires is an example of stiff systems that are well-known to be numerically unstable when the strain becomes too high. For instance, a wire with segment length of the order of $1\ m$, particle mass $1\ kg$ integrated with the symplectic Euler algorithm at step-size $h = 1/60s$ cannot support loads much larger than $100\ kg$ in normal gravity. This is a severe limitation in simulations where the wires and cables should be used for heavy hoisting or anchoring. The instability may also develop in the absence of heavy loads by the wire inertia of its own, e.g., in a whip effect. The cause of the numerical instability is that when the wire has normal modes of transversal vibrations with frequencies $\omega \geq h^{-1}$. These modes are excited by the numerical noise in the system. The normal mode frequencies are proportional to the wave velocity of transversal vibrations $c \equiv [fL/m_{\text{tot}}]^{1/2}$, where f is the wire tension, $m_{\text{tot}} = Nm$ is the total wire mass and wire length L. With increasing wire tension the transversal force component increases, see Fig. 2, and the normal mode oscillation frequencies increases along with it. The spectrum of normal modes for a wire of homogeneous distribution of N particles is discrete and the normal mode frequency ω_n ranges [Fetter and Walecka 1980]

$$\omega_n = \frac{2(N+1)c}{L} \sin\left[\frac{n\pi}{2(N+1)}\right] \quad , \; n = 1, 2, \ldots, N \tag{5}$$

The maximum frequency is $\max(\omega_n)$ can be approximated to $2(N{+}1)c/L$. A necessary condition on the number of point masses for the wire to be numerically stable can thus be formulated as

$$N < N_{\text{crit}} \equiv \frac{1}{2h}\sqrt{\frac{Lm_{\text{tot}}}{f}} \tag{6}$$

Observe that the extreme case with $N = 0$ is unconditionally stable. This is the *massless cable* in [Servin and Lacoursière 2007] and has no modes of transversal oscillations. If the wire would have been simulated with a time-explicit integrator and an explicit force model such as a spring forces the more restrictive *Courant condition* for numerical stability should have been included as well.

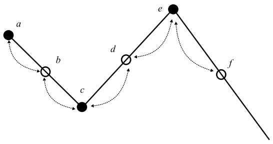


Figure 3: *Illustration of the* COARSE/REFINE*-transition for adaptive resolution of wires. In order to preserve total mass and momentum these are redistributed over the neighboring particles.*

3.1 Adaptive wire resolution

In order to realize an implementation of the adaptive resolution algorithm, Algorithm 1, we must decide on quality measures for the wire systems, define a wire resolution number and construct transition rules for wire refinement and coarsening.

Just for the clarity of the presentation here we assume spatially homogeneous distribution of wire masses and let the number of masses depend on a *resolution number* $r = 0, 1, 2, 3, \ldots$, as $N = 2^r + 1$. Each *refinement*, $r \rightarrow r_+ = r + 1$, and coarsening, $r \rightarrow r_- = r - 1$, means the number of wire segments are doubled or halved, respectively, as depicted in Figure. 3. The transition to another level of resolution should preserve total wire mass, local center of mass, local wire rest length and the linear momentum. The following transition rules accomplishes that. The particle mass depends on the resolution number as $m_{\text{tot}}/(2^r + 1)$. New particles b are added at half distance along the wire segments linking existing particles a and c. After eliminating a particle b the new wire segment is the straight wire connecting the two neighboring particles a and c. A REFINE-transition transforms the velocities as $(\dot{q}_a, \dot{q}_c, \dot{q}_e, \ldots) \rightarrow (\dot{q}_{a+}, \dot{q}_{b+}, \dot{q}_{c+}, \dot{q}_{e+}, \dot{q}_{f+}, \ldots)$, where

$$\dot{q}_{a+} = \dot{q}_a \tag{7}$$

$$\dot{q}_{b+} = \left(\tfrac{2^{r+1}+1}{2^r+1} - 1\right)\left[\tfrac{1}{\eta_a}\dot{q}_a + \tfrac{1}{\eta_c}\dot{q}_c\right] \tag{8}$$

$$\dot{q}_{c+} = \dot{q}_c \tag{9}$$

where η_a and η_c are the number of neighbors (1 or 2) of particle a and b, respectively. A COARSE-transition transforms the velocities as $(\dot{q}_a, \dot{q}_b, \dot{q}_c, \dot{q}_d, \dot{q}_e, \dot{q}_f, \ldots \rightarrow (\dot{q}_{a-}, \dot{q}_{c-}, \dot{q}_{e-}, \ldots)$, where

$$\dot{q}_{c-} = \tfrac{2^{r-1}+1}{2^r+1}\left[\dot{q}_c + \tfrac{1}{2}\left(\dot{q}_b + \dot{q}_d\right)\right] \tag{10}$$

with the exception for particles at the end points of the wires, in which case the contribution from neighboring particles—the b-term or the d-term in Eq. (10)—vanishes. Coarsening of a curved wire may produce significant compression of wire segments and violation of preservation of the total wire length. Depending on how these potentially large constraint violations are treated this may cause large energy injections and give rise to jittery and instabilities. We avoid this by allowing wire compression through modification of the constraint to an inequality constraint $g^{wire} \geq 0$. It can be shown that these transitions preserve the total momentum of the wire.

We define the wire system quality measure to be $Q =$

$(Q_{\text{acc}}, Q_{\text{time}}, Q_{\text{rob}})$ with

$$Q_{\text{acc}} \equiv \left(\frac{N}{N_p}\right)^{\gamma_{\text{acc}}} \tag{11}$$

$$Q_{\text{time}} \equiv 1 - \frac{1}{1 + e^{\gamma_{\text{time}}(t_{\text{lim}} - t_{\text{est}})}} \tag{12}$$

$$Q_{\text{rob}} \equiv 1 - \frac{1}{1 + e^{\gamma_{\text{rob}}(N_{\text{crit}} - N)}} \tag{13}$$

We do not claim that this measure of quality is unique nor canonical, but it has the functional dependency required for adaptive resolution for optimal quality in fixed time-step realtime simulations. The γ-exponents control the sensitivity to variations in N. For simplicity we have left out Q_{user} from the measure. The effect of user view and interaction are instead incorporated by modifying the weight factor w_{acc}. In practice, you may need to regularize the quality measure Q by adding a small positive number to avoid division by zero when the cost function is computed.

The COARSE-REFINE-algorithm for adaptive resolution for optimal wire quality we use is given by Algorithm 2. The computation of the wire tension required for line 7 in the algorithm is computed from the wire constraint force $G^T\lambda$.

Algorithm 2 Adaptive wire resolution

```
1:  system initialization $(q, \dot{q}, g, r)$
2:  while VE running do
3:      user interaction $\rightarrow (w_{\text{acc}}, t_{\text{lim}})$
4:      accumulate explicit forces $-\nabla_q V$
5:      update contact data and constraint data $g$
6:      step particles
7:      compute quality measure $(Q_{\text{acc}}, Q_{\text{time}}, Q_{\text{rob}})$
8:      compute optimal wire resolution
        $r' = \arg\min_r \frac{1}{2} \sum_\alpha w_\alpha Q(r)_\alpha^{-2}$
9:      if $r' < r$ then
10:         while $r' < r$ do
11:             COARSE$\rightarrow (q_-, \dot{q}_-, g_-, r - 1)$
12:         end while
13:     end if
14:     if $r' > r$ then
15:         while $r' > r$ do
16:             REFINE$\rightarrow (q_+, \dot{q}_+, g_+, r + 1)$
17:         end while
18:     end if
19:     Set new particle mass $m = m_{\text{tot}}/(2^n + 1)$
20: end while
```

3.2 Numerical experiments

The implementation of the adaptive wire model is made using the SPOOK-stepper introduced in [Lacoursière 2007] implemented in MATLAB. We describe the implementation in professional software in Sec. 3.3

3.2.1 Accuracy, stability and realtime performance

In order to test and demonstrate the presented method we set up a system consisting of a box of mass 100 kg supported by a wire of total mass 1 kg and length 10 m. Gravity is set to 9.8 m/s^2 and the time step $h = 1/60\ s$. This system is unstable at mass ratios $1/1000$ and above, i.e., for wire resolution at $\gtrsim$ 10 particles under the weight of the box. The system is unstable for even finer resolution when the wire tension peaks, say, after a dropping the box from above. We let the resolution number and number of wire masses range between $(r, N) \in (0, 2), (1, 3), (2, 5), (3, 9), (4, 17), (5, 33)$. The parameters of the quality functions are $w_{\text{acc}} = w_{\text{time}} = w_{\text{rob}} = 1$, and $\gamma_{\text{acc}} = 2$, $w_{\text{time}} = 100/h$, $\gamma_{\text{time}} = 10$. The simulation is run for 8 $seconds$ and the result is displayed in Fig. 4, showing a series of snapshots and the time evolution of the quality measures and the number of wire particles. It can be seen that the wire avoids instability by decreasing the resolution at the high tension phase at times $t = 0 - 0.5\ s$, $t = 3.5 - 4.5\ s$ and $t = 7 - 8\ s$, and maximizes the resolution at the turn points at times $t = 2\ s$ and $t = 6\ s$. Between time t = 5–7 s we have inserted a dip in the available computational time t_{lim} that makes the resolution algorithm to decrease the resolution to maintain realtime performance.

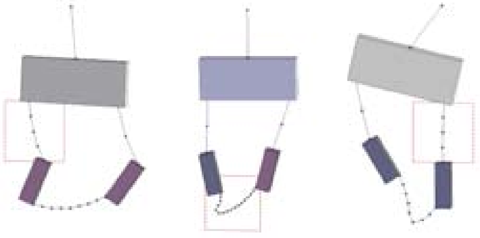


Figure 5: *Snapshots from simulation of several boxes and wires with view-dependent wire resolution. The view area is marked with red dashed wire and focuses at different wires at different times and thereby redistributes the available computational time to keep this wire at higher resolution if this is permitted by the stability requirement.*

The gain of the method can be understood from the following data from the numerical experiments. To have stable simulation at full wire resolution of the system and scenario described here would require a time step 20 times smaller than $h = 1/60\ s$ and consume increased computational power of *at least* the same factor. The alternative would be to keep the resolution at a level guaranteed to be stable at all times – in this case this would be at zero resolution ($r = 0$) – which would give poor accuracy and visual appearance. The computational overhead of the adaptive resolution algorithm for wires is small, roughly 1% of the total computational time spent on computing the dynamics

3.2.2 View-dependent adaptivity

We now extend the system of Sec. 3.2.1 to a system with four wires each of total mass 1 kg connecting three rigid boxes with masses $m_1 = 10\ kg$ and $m_2 = m_3 = 1\ kg$. The estimated computational time is $t_{\text{est}} = \sum_{i=1,2,3,4} t_{\text{est},i}$. The view-dependency is introduced by increasing the accuracy weight factor $w_{\text{acc},i}$ of the wire “in view” by a factor 10^3. The adaptive wire resolution algorithm responds by giving the system “in view” higher resolution and more computational time, as long as this does not threaten the stability or realtime performance requirements. Snapshots from a simulation with view-dependent resolution are displayed in Fig. 5. The wire “in view” is the one marked by the red dashed box.

3.3 Implementation of adaptive wires in simulator software

Next we describe the implementation of adaptive wire resolution in a 3D physics simulation library called AgX [AgX] and list some important conclusions from this development. The AgX library was

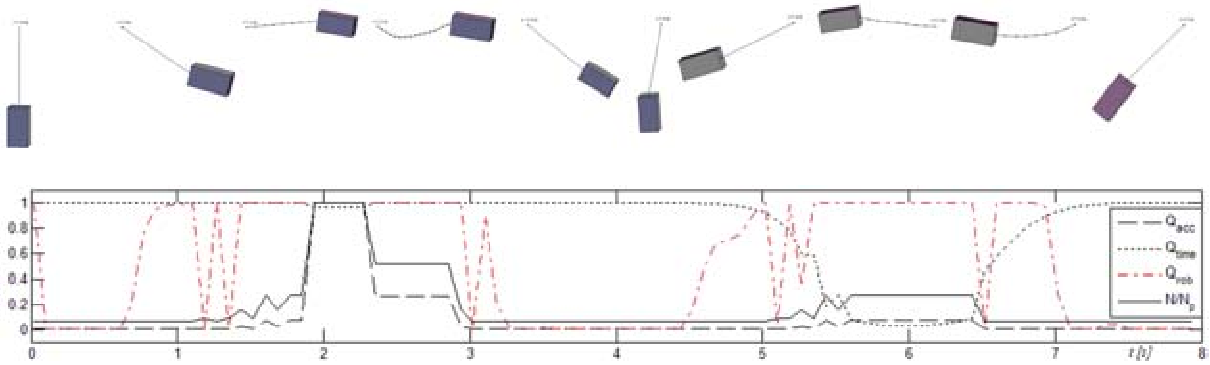


Figure 4: *The time evolution of a rigid box supported by a wire with adaptive resolution. The upper sequence of figures shows the system at different times – with the wire resolution varying with the stability criterion at high wire tension. The lower figure shows the wire quality measures and the number of wire masses (normalized by N_p) as functions of time. About the time $t = 6$ s there is a dip in the available computational time which forces the system to be coarsened.*

crafted for the realisation of professional simulators, e.g., training simulators for operation of heavy vehicles and ships. The implementation differs in several ways from the method presented in the present paper. Most importantly, it shares the strategy of securing numerical robustness by recognizing unstable states and simplifying the system to higher resolution for which the system is stable. One of the differences is that that it uses local resolution and inhomogeneous mass distribution instead of homogeneous and global wire resolution. The features of this wire model, which will be covered in more detail in a separate publication (in preparation), includes: *i) Slide nodes.* Besides attachment nodes at the ends of the wires, we include *massless slide nodes* as described in [Servin and Lacoursière 2007]. A rigid body attached to a wire with a slide node can slide along the wire like a "bead on a wire". *ii) Frictional contact nodes.* Contact nodes are a natural extension of slide nodes. Instead of having body fixed position the contact nodes are continuously updated to be at the position on the body surface that minimizes the wire segment length. If the force at the contact node is directed "outwards" from the body the node is eliminated and the wire contact is thus detached. Stick friction in the wire direction is modeled by treating the contact node as an attachment node if the node force is within the friction cone. Sliding friction over the body surface is introduced with a friction parameter between zero and unity that diminishes the movement of the contact node by this factor. *iii) Wire elasticity.* Elasticity with respect to stretching and bending deformations of the wire is simulated following the method presented in [Servin and Lacoursière 2008] whereby established material models, e.g., for steel wire ropes and chains, can be used within the framework of constrained multibody systems.

In the development and implementation of this wire model we have found that adaptive resolution is a key element to make the wires numerically robust. The passage of wire masses over slide nodes and contact nodes is automatically handled by the resolution algorithm which eliminates the particles as they approach the node. Without this feature the local oscillation mode frequency would peak as the node and particle comes close together and the wire would become unstable. As a bonus, no special method to handle the passage of particles over nodes without risk of reflections needs to be implemented. Furthermore, handling the contacts by massless contact nodes rather than by light wire masses makes it possible to change resolution without introducing fluctuating contact forces and jittery particle behavior. The title figure, Fig. 1, shows an image from one of the test scenes. This system includes two boxes connected by a wire running over two rigid cylindrical wheels. Black dots shows lumped wire masses and magenta dots show massless contact nodes and their neigbouring wire mass nodes also involved in the friction model. Pulling a box makes the wheel turn by the wire-wheel friction force. Running the wheels with a motor makes them pull the wire and the boxes along with them. If the friction force is too small the wire will slide over the surface of the wheels. If the tension becomes large the system is simplified and with high enough tension no wire masses remains, only contact nodes and end attachment nodes. In Fig. 6 we show an image from a test-scene in the development of training simulators for anchor handling ship. These training scenarios involve multiple wires of length ranging from a few meters to several kilometers, complicated wire routings connecting various ship parts, other ships, oil rig and heavy anchors. The wire is connected to a drum and motor and has frictional contacts with the ship, in particular with guide pins and the stern roller at back of the ship.

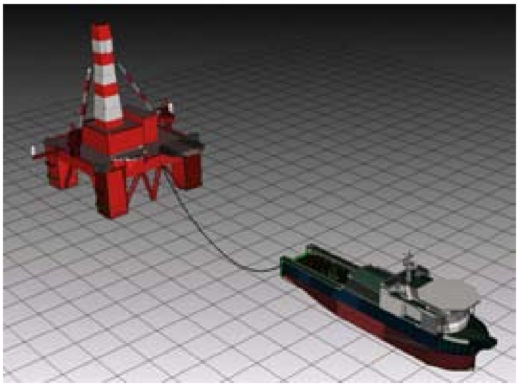


Figure 6: *Snapshot from a test-scene of the implementation of wire with adaptive resolution in a physics library used for training simulators. Wires occurs in ship handling oil rig anchoring. The wire routing can be complicated, connecting ship parts with oil rig, anchors and seabed. The wire length may vary up to several kilometers and sometimes be at very large tension.*

4 Summary and discussion

We have presented a systematic approach to adaptive resolution in physics based virtual environments. The resolution adapts to maintain optimal quality with respect to a number of quality requirements imposed on the application, e.g., realtime performance, visual appearance, accuracy and numerical robustness. The latter two requirements are specific for physics based applications and are not included in conventional level-of-detail algorithms. We have presented a specific realization of this idea to multibody simulation involving wire systems and demonstrated how this improves the robustness, visual quality and time efficiency. Numerical experiments (Sec. 3.2.1) confirm that the overall computational performance can be improved by several orders in magnitude, as compared to resolving the instabilities by smaller integration time steps. We briefly described the implementation of this model in a software library used in commercial off-shore training simulators. An important conclusion from this development is adaptive resolution is a key element also to obtain robust simulation of wires with frictional contact nodes and slide nodes.

The computational power of personal computers is steadily increasing and seems to continue to do so with the emerging *massive multicore* CPU architecture. Adaptive resolution techniques is one important ingredient in order to gain the full potential of the increased computing power in VEs. Adaptive resolution of *physical systems* is particularly difficult. The most critical part is adaption into more coarse grained models in order to improve numerical robustness and diminish computational time. The general algorithm requires that the sub-systems of the VE are provided with graphical and physical models in a spectrum of granularity ranging from high to low. At lowest granularity the models should be simple model primitives, such as single rigid bodies or particles, that can be simulated extremely stable and fast. Complex objects important to the application, like a vehicle, can then be given a hierarchy of models ranging from a single primitive, to multiple connected rigid bodies and up to highly fine-grained models including deformable parts (tires, antenna, steel body, etc) and mechanical details (doors, suspension system, mechanical parts of the drive line, etc). This hierarchy of models should be assigned with unique resolution numbers, transition rules for refinement or coarse graining during simulation and quality measures with respect to numerical stability, computational time, accuracy and visual appearance. The adaptivity resolution algorithm will then automatically adapt the system to optimal resolution. If a subsystem approaches numerical instability or has its computational time decreased it will be substituted by a simplified system and more robust system. Much work remains to be done in this area, e.g., supply methods for predicting the numerical stability of complex systems at different resolution. Even though the more complex models are not as easily analyzed as wire systems they have normal modes of oscillations with some frequency spectrum depending on the resolution and occurrence of light elements and high tension. More coarse grained system has lower frequency spectrum and is more stable. Stability predictors can be constructed from known instability mechanisms. An alternative is to do precomputation of systems at different resolution levels and build prediction models in form of tables.

References

AGX. Agx multiphysics toolkit. `http://www.algoryx.se/`.

CARLSON, D. A., AND HODGINS, J. K. 1997. Simulation levels of detail for real-time animation. In *Proceedings of the conference on Graphics interface '97*, Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 1–8.

CHENNEY, S., AND FORSYTH, D. 1997. View-dependent culling of dynamic systems in virtual environments. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, ACM, New York, NY, USA, 55–58.

CLARK, J. H. 1976. Hierarchical geometric models for visible surface algorithms. *Commun. ACM 19*, 10, 547–554.

DEBUNNE, G., DESBRUN, M., CANI, M.-P., AND BARR, A. H. 2001. Dynamic real-time deformations using space & time adaptive sampling. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, E. Fiume, Ed., ACM SIGGRAPH, 31–36.

FETTER, A., AND WALECKA, J. D. 1980. *Theoretical Mechanics of Particles and Continua*. McGraw-Hill, 108–119.

GILLILAN, R. E., AND WILSON, K. R. 1992. Shadowing, rare events, and rubber bands - a variational Verlet algorithm for molecular-dynamics. *J. Chem. Phys. 97*, 3, 1757–1772.

KHAREVYCH, L., YANG, W., TONG, Y., KANSO, E., MARSDEN, J. E., SCHRÖDER, P., AND DESBRUN, M. 2006. Geometric, variational integrators for computer animation. In *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, ACM SIGGRAPH/Eurographics, 43–51.

LACOURSIÈRE, C. 2007. *Ghosts and Machines: Regularized Variational Methods for Interactive Simulations of Multibodies with Dry Frictional Contacts*. PhD thesis, Department of Computing Science, Umeå University, Sweden, SE-901 87, Umeå, Sweden.

REDON, S., GALOPPO, N., AND LIN, M. C. 2005. Adaptive dynamics of articulated bodies. ACM, New York, NY, USA, vol. 24, 936–945.

SERVIN, M., AND LACOURSIÈRE, C. 2007. Massless cable for real-time simulation. *Computer Graphics Forum 26*, 2, 172–184.

SERVIN, M., AND LACOURSIÈRE, C. 2008. Rigid body cable for virtual environments. *IEEE Transactions on Visualization 14*, 4, 783–796.

SPILLMANN, J., AND TESCHNER, M. 2008. An adaptive contact model for the robust simulation of knots. *Computer Graphics Forum 27*, 2, 497–506.

Enhanced Interactive Spiral Display

Christian Tominski[*]
Institute for Computer Science
University of Rostock

Heidrun Schumann[†]
Institute for Computer Science
University of Rostock

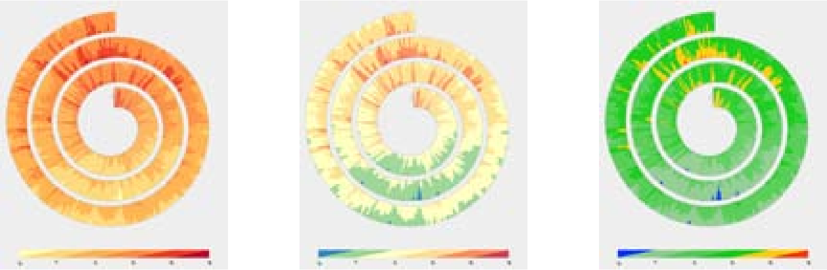


Figure 1: *Enhanced interactive spiral display – Left to right: Red sequential color scale, rainbow color scale, and custom color scale*

Abstract

Spiral displays have been developed to visualize the cyclic character of a data set. They are particularly useful for spotting seasonal patterns in time series data. This, however, requires expressive visual encoding and efficient interaction.

In this paper, we improve existing spiral displays by applying the expressive two-tone pseudo coloring. This allows users to read off data values more precisely. More importantly, we enhance our two-tone spiral display with efficient interaction facilities. These allow for easy exploration of the data space and easy adjustment of a variety of visualization parameters. Our tool is fully implemented and freely available for experimentation.

CR Categories: H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical User Interfaces, Interaction Styles; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction Techniques

Keywords: Information visualization, interaction, spiral display

1 Introduction

Finding seasonal patterns is a common task when analyzing time-oriented data. Analytic methods can be applied to find such patterns. However, results depend heavily on the analysis method used, and choosing the right method for a particular application is not always an easy task. Especially in cases where no or only vague hypotheses about the data exist, it is difficult to decide which automatic analysis methods might work. This is where interactive visual exploration comes into play. Visual methods can support analysts in exploring unknown data and generating hypotheses about them. Moreover, a visual representation can help in confirming results of analytic procedures or simply support the communication of analysis results to a broader audience.

In this paper we focus on spiral displays. Such representations have been shown to be helpful tools to spot cyclic pattern in time series data [Aigner et al. 2008]. In order to confirm or communicate analysis results, an expressive visual encoding is required. Classic spiral displays rely on simple color coding or glyphs. Moreover, efficient interaction facilities are required to support easy data exploration. [Aigner et al. 2008] explain that previously unknown cyclic patterns can be found with the help of interaction. But except for the use of common control panels, only little has been reported on how to interact with spiral displays.

We enhance classic spiral displays with respect to both visual encoding and interaction. We apply two-tone pseudo coloring, an expressive visual mapping that provides overview and precise detail at the same time. Our spiral displays is not a static display, but provides a variety of ways of navigating the data and adjusting the visual representation. We believe that our two-tone spiral display is an interesting and easy-to-use alternative to existing approaches.

In the next section, we will take a brief look at previous work. In Section 3, we focus on the visual aspects of our approach. Aspects of interaction are described in Section 4. Finally, we conclude and present ideas for future work in Section 5.

2 Related Work

Cyclic representations of data are helpful in many application scenarios. [Harris 1999] presents several cyclic data drawings. Among them are spiral displays. Compared to other cyclic data representations, spirals have the advantage that they can encode both cyclic and linear aspects of the data. Cyclic aspects are encoded to the cycles of a spiral. Since the order of cycles is preserved in a spiral it is possible to mentally unroll a spiral to understand linear aspects.

Spiral displays have a long history as visual depictions. Nowadays, several computer implementations of spiral displays are known in the literature.

[Carlis and Konstan 1998] present a first prototype that uses spirals as the central element of the visualization. The spiral is used to arrange visual items, which can be dots, cans, or simple bar charts. The size of these items encodes data values.

In [Hewagamage et al. 1999], the 3D form of a spiral, that is, a helix, is used to visualize spatio-temporal patterns on maps. To this end, 3D helices are placed on a 3D map display. Attached to the helices are small colored icons that represent events in the visualized time series. In order to focus on certain alignments of events, it is possible to vary the cycle length within a helix.

[*] e-mail: ct@informatik.uni-rostock.de
[†] e-mail: schumann@informatik.uni-rostock.de

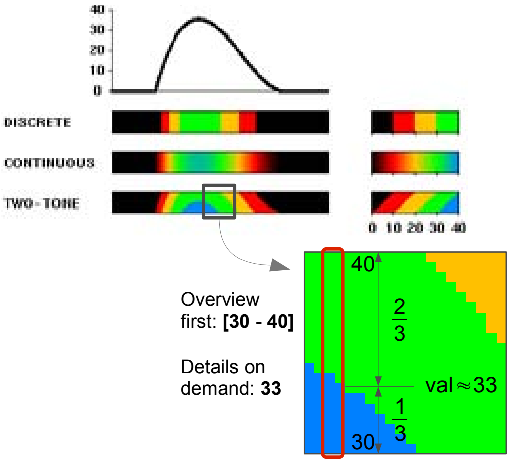


Figure 2: *Two-Tone Pseudo Coloring explained.*

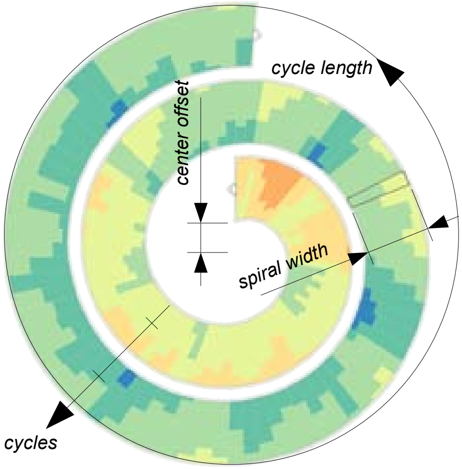


Figure 3: *Parameters of the spiral construction.*

Another use of spirals is presented by [Ward and Lipchak 2000]. They describe a tool for visualizing multivariate data. The approach is to use glyphs that encode multiple variables and to arrange the glyphs on the screen. Besides two other glyph layouts, a spiral layout is provided to emphasize the cyclic character of the data.

The spiral described by [Weber et al. 2001] uses color and line thickness to encode time series data. In order to visualize more than one variable, multiple spirals can be intertwined. An interesting concept to cope with larger time series is to combine the 2D spiral with a 3D helix.

[Dragicevic and Huot 2002] apply the spiral shape quite differently than the previously mention approaches. They combine the spiral with a clock metaphor to construct a so called SpiraClock. It is a very helpful tool to visualize upcoming events in time.

Similar to [Hewagamage et al. 1999] is the approach of [Tominski et al. 2005]. Here, the goal is to visualize space- and time-dependent data. Besides pencil glyphs for linear data, helix glyphs are offered for cyclic data. The helices use color coding and can be split into sub-bands to allow for visual comparison of multiple variables.

All these approaches underline the usefulness of spiral displays for data visualization. In the following, we will further improve them with respect to visual encoding and interactivity.

3 Two-Tone Spiral Display

In information visualization color is a very prominent visual variable. Color coding is effective for qualitative as well as quantitative data. However, discerning data values precisely is not always easy for users. Figure 2 shows examples of discrete and continuous color coding. With a discrete color scale, only ranges of colors, and hence, only ranges of data values can be identified. In contrast to that, continuous color coding is based on a scale of smooth color transitions. However, users tend to perceive a sequence of colors as a set of discrete ones [Ware 2000]. So, even with very smooth transitions, values cannot be read very accurately.

Two-Tone Pseudo Coloring (TTPC) has been introduced to address this issue [Saito et al. 2005]. TTPC uses a discrete color scale that consists of only a few colors (less than eight). In contrast to classic color coding (discrete or continuous) where each data value is mapped to one color, TTPC uses two colors being adjacent in the color scale to encode a data value. These two colors are interpreted in two steps (see Figure 2). First, the two colors guide users to a particular interval in the value range. If users find that interval interesting, they go into detail: The proportion of use of both colors encodes the precise data value. This two-step interpretation is the basis for overview+detail. While the first step relies on the spontaneous perception of color for the overview, the second step is based on the human capabilities to judge lengths, which provides for sufficient detail. Compared to other approaches, TTPC generates very compact, yet precise visual representations.

TTPC in its original form is suited to represent univariate data with linear character. We have previously married TTPC with the Table Lens approach to allow for multivariate data visualization [John et al. 2008]. However, cyclic representations using TTPC have not yet been reported on, even though the inventors of TTPC indicate that it is possible to apply TTPC to non-linear geometry.

In order to apply TTPC to a spiral display, we have to discretize the continuous shape of the spiral into cells each representing a single data value. The cells are then colored by applying TTPC. There are four parameters to control the construction of spiral cells. The first two are *cycle length* and *cycles*, which determine the number of data values to be mapped per spiral cycle and the number of cycles of the spiral, respectively. Because the relation of two colors has to be judged per cell, cells must have a certain extend. This extend is controlled via the parameter *spiral width*. Since cells become infinitely thin in the very center of the spiral, it makes sense to move the starting point for cells away from the center. This way we assure that cells can still be recognized well. The amount of translation from the center is linked to the parameter *center offset*. Figure 3 illustrates the spiral construction and the parameters involved. All these parameters, but particularly *cycle length* and *cycles*, influence the visual representation and the conclusions that might be drawn from it. This is why providing easy to use interaction methods for parameter adjustments is so important (see Section 4).

Each cell of the spiral uses TTPC to encode data. In a first step, it is necessary to determine the two colors required for the data value to be represented in a cell. Secondly, the spiral cell is divided into two parts whose sizes are computed based on TTPC. Finally, both parts are colored. We provide several predefined color scales,

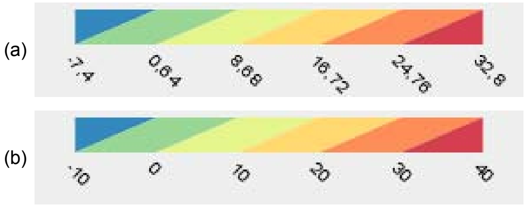


Figure 4: *Color mapping – (a) Standard mapping using* $min = -7.4$ */* $max = 32.8$ *as found in the data; (b) Mapping with extended value range of* $min = -10$ */* $max = 40$.

which we derived from ColorBrewer [Harrower and Brewer 2003]. In addition to that, users can define their own custom color scale or adjust the existing ones (see next section). How different the visual results might be when using different color scales is demonstrated in Figure 1. The figures shows average temperatures in the city of Rostock for a period of three years, roughly a thousand data values. On the left, a red sequential color scales provides an overview of the value distribution. The rainbow color scale used for the spiral in the middle helps to discern regions in the data. On the right hand side, the user has customized the color scale to emphasize rather cold days (blue) and quite hot days (orange). In all cases, TTPC encoding facilitates overview and provides detail when needed.

4 Interactive Data Exploration

We implemented the TTPC spiral as an interactive tool that allows for manifold interaction with the data and the visual representation.

Labeling As an elementary interaction, our tool allows picking of spiral cells in order to provide labels showing data values in textual form. Upon user request, we offer a global label that shows start time and end time of the spiral to provide better orientation during browsing in time (as described next).

Browsing in time Since spiral displays address time series data, our implementation has to support browsing in time. In contrast to other tools, we use direct manipulation for that purpose. Two alternatives are provided to move in time, if not all data values fit the spiral (this depends on the size of the data set and on the parameters *cycle length* and *cycles*). The first option is to use the two small arrow-shaped buttons that are shown at each end of the spiral. Pressing such a button starts browsing through time in the selected direction. The longer a button is pressed, the faster the browsing will be. Alternatively, users can drag the mouse away from the button to control the browsing speed. The second option to browse in time is to use the keyboard. Pressing UP or DOWN keys moves one time step forward or backward in time. To move faster, users can also use PAGE UP and PAGE DOWN keys or the mouse wheel, which moves the current view a full spiral cycle in the desired direction. While mouse interaction is helpful for roughly browsing the data, keyboard controls are better suited for fine adjustments or to skip larger regions in the data. In conjunction, mouse and keyboard controls allow for efficient browsing in time.

Overview and detail By adjusting the parameters *cycle length* and *cycles*, users can determine the number of data values to be mapped to the spiral display. Apparently, larger numbers of data values provide overview, but in order to see detail only a few data values should be shown. By utilizing the browsing functionality, users can switch between different parts of the data to see detail.

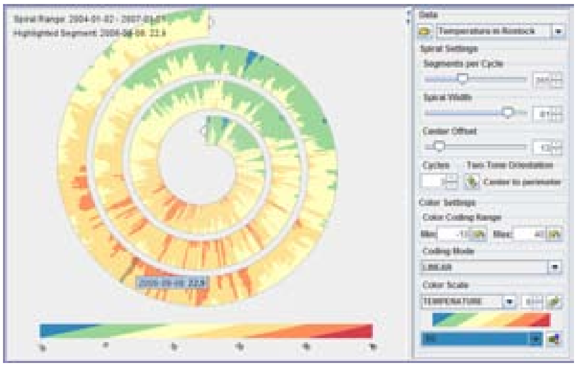


Figure 5: *The user interface – The central view shows the spiral visualization and the color legend. All visualization parameters can be adjusted in the settings panel on the right.*

However, adjusting parameters and performing browsing can be inconvenient when switching frequently between overview and detail. Therefore, we offer an alternative again based on direct manipulation. By simply dragging the mouse, users can select a subset of the data for further detailed investigation. During dragging, a frame marks the selected cells. In detail mode, only the selected cells are shown. Users can then decide to continue selecting an even smaller subset of the data or return to the overview via right click. This way of switching between overview and detail requires only a few clicks and is much easier than manipulating parameters.

Color scale adjustment The way of how colors are used affects what can be seen from the visualization [Silva et al. 2007]. Therefore, it is mandatory to provide mechanisms that allow users to adapt the color coding to their particular needs and tasks. Our spiral tool supports several adjustments. The first one is to adjust the mapping range. Usually, colors are mapped based on the value range between minimum and maximum data value. However, the resulting color legend is difficult to interpret in most cases (see Figure 4(a)). A simple possibility to overcome this problem is to allow users to extend the value range for the color mapping. By choosing minimum and maximum carefully, the legibility of the color legend can be improved significantly (see Figure 4(b)).

Further ways of manipulating the color mapping include specifying the number of segments in the color scale, editing particular colors, flipping the color scale, and switching between linear and exponential mapping functions. The latter is particularly useful for data with skewed value distributions. The possibility to choose from a set of predefined color scales has already been mentioned earlier.

User interface The user interface is crucial when it comes to applying a visualization method in various scenarios. We opted for a layout that presents the spiral as the central view with a color legend on the bottom. A panel for all adjustable parameters is presented to the right of the spiral. The panel encapsulates related parameters in groups to create order among the various possibilities of adjustment. Once the user has tuned the parameters to the task at hand, the settings panel can be hidden away to fully concentrate on the visualization. Figure 5 shows the user interface of our spiral tool.

Example: How to find a cyclic pattern? We will now give a brief example of how interaction helps in finding a cyclic pattern in a human health data set. Figure 6(a) shows a spiral where the

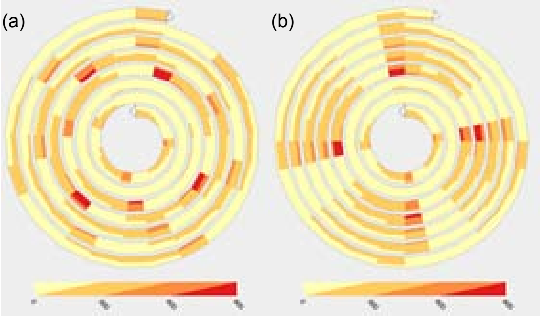


Figure 6: *Finding a pattern – (a) Cycle length* = 25*; (b) Cycle length* = 28

cycle length is set to 25 days – a clear pattern can not be seen. After adjusting the *cycle length* to 28 days in Figure 6(b) a pattern becomes obvious: It is a weekly pattern of high numbers of new infections of the respiratory reported in the beginning of a week gradually decreasing to no reports on weekends. Indeed, this is a simple pattern, but still the example demonstrates how interaction with the visual representation facilitates finding interesting things in the data.

5 Preliminary Results and Future Work

In this paper, we have presented an enhanced interactive spiral display. The novelty of our approach lies in the application of Two-Tone Pseudo Coloring (TTPC) to non-linearly shaped geometry, the spiral in our case. This combines the advantages of TTPC and spiral displays, and thus potentially better facilitates exploration of cyclic patterns in time-oriented data. A major objective was to focus on interactivity, which we think is the important aspect of data exploration. For that reason, we provide easy to use mechanisms for browsing in time and for switching between overview and detailed views. Over existing spiral displays, our tool has the advantage that it is based on direct interaction with the visual representation. More interaction, particularly adjustment of the spiral shape and manipulation of the color mapping, is possible via a well-chosen user interface. Our approach is available as a fully implemented tool, which can run either as stand-alone application or as a java applet in a web browser [Tominski et al. 2007].

Even though the spiral display we described is quite capable, there are many things that can be improved. First of all, our spiral needs to be extended to allow for multivariate data exploration. This can be achieved by intertwining multiple spirals. Secondly, even though interaction is a very powerful means to allow users to explore all aspects of a data set, relying solely on the users' capability to find suitable parameter settings for cycle lengths or colors might not always be the best solution. It makes sense to provide guidance or to make suggestions on how to set parameters to reach certain goals. Such an extension could be driven by utilizing appropriate analysis methods to find potential cyclic patterns. For more complex patterns, it might also be interesting to experiment with spirals using irregular cycle lengths, i.e., each cycle might show a different number of data elements. For adjusting the color scale automatically, one could apply the task-driven methods described in [Tominski et al. 2008].

Last but not least, we need to collect feedback from users to find possible weaknesses of our approach and to expand its strengths. We encourage users of our applet to send us their feedback.

Acknowledgments

We thank Prof. Takafumi Saito for support and gratefully acknowledge the help of Steffen Hadlak in implementing the spiral tool.

References

AIGNER, W., MIKSCH, S., MLLER, W., SCHUMANN, H., AND TOMINSKI, C. 2008. Visual Methods for Analyzing Time-Oriented Data. *IEEE Transactions on Visualization and Computer Graphics 14*, 1.

CARLIS, J. V., AND KONSTAN, J. A. 1998. Interactive Visualization of Serial Periodic Data. In *Proc. of ACM Symposium on User Interface Software and Technology (UIST'98)*, ACM Press.

DRAGICEVIC, P., AND HUOT, S. 2002. Spiraclock: A continuous and non-intrusive display for upcoming events. In *Extended Abstracts of ACM Conference on Human Factors in Computing Systems (CHI'02)*, ACM Press.

HARRIS, R. L. 1999. *Information Graphics: A Comprehensive Illustrated Reference*. Management Graphics, Atlanta.

HARROWER, M. A., AND BREWER, C. A. 2003. ColorBrewer.org: An Online Tool for Selecting Color Schemes for Maps. *The Cartographic Journal 40*, 1.

HEWAGAMAGE, K., HIRAKAWA, M., AND ICHIKAWA, T. 1999. Interactive Visualization of Spatiotemporal Patterns Using Spirals on a Geographical Map. In *Proc. of IEEE Symposium on Visual Languages (VL'99)*, IEEE Press.

JOHN, M., TOMINSKI, C., AND SCHUMANN, H. 2008. Visual and Analytical Extensions for the Table Lens. In *Proc. of IS&T/SPIE Visualization and Data Analysis (VDA'08)*, SPIE Press.

SAITO, T., MIYAMURA, H. N., YAMAMOTO, M., SAITO, H., HOSHIYA, Y., AND KASEDA, T. 2005. Two-Tone Pseudo Coloring: Compact Visualization for One-Dimensional Data. In *Proc. of IEEE Symposium on Information Visualization (InfoVis'05)*, IEEE Press.

SILVA, S., MADEIRA, J., AND SANTOS, B. S. 2007. There is More to Color Scales than Meets the Eye: A Review on the Use of Color in Visualization. In *Proc. of International Conference Information Visualisation (IV'07)*, IEEE Press.

TOMINSKI, C., SCHULZE-WOLLGAST, P., AND SCHUMANN, H. 2005. 3D Information Visualization for Time Dependent Data on Maps. In *Proc. of International Conference Information Visualisation (IV'05)*, IEEE Press.

TOMINSKI, C., HADLAK, S., AND SCHUMANN, H., 2007. Two-Tone Pseudo Colored Spiral Display. Java Applet. `http://vcg.informatik.uni-rostock.de/~ct/TTS/TTS.html` (accessed Sept. 2008).

TOMINSKI, C., FUCHS, G., AND SCHUMANN, H. 2008. Task-Driven Color Coding. In *Proc. of International Conference Information Visualisation (IV'08)*, IEEE Press.

WARD, M. O., AND LIPCHAK, B. N. 2000. A Visualization Tool for Exploratory Analysis of Cyclic Multivariate Data. *Metrika 51*, 1.

WARE, C. 2000. *Information Visualization: Perception for Design*. Morgan Kaufmann, San Francisco.

WEBER, M., ALEXA, M., AND MLLER, W. 2001. Visualizing Time-Series on Spirals. In *Proc. of IEEE Symposium on Information Visualization (InfoVis'01)*, IEEE Press.

Computation of Topologic Events in Kinetic Delaunay Triangulation using Sturm Sequences of Polynomials

Tomáš Vomáčka[*]
Ivana Kolingerová[†]
University of West Bohemia

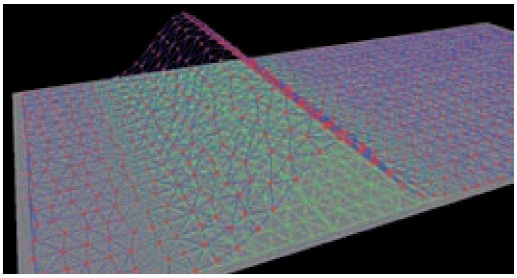


Figure 1: *Experimental implementation of the kinetic Delaunay triangulation for 2.5D wave simulation.*

Abstract

Even though the problem of maintaining the kinetic Delaunay triangulation is well known, this field of computational geometry leaves several problems unsolved. We especially aim our research at the area of computing the times of the topologic events. Our method uses the Sturm sequences of polynomials which, combined together with the knowledge in associated field of mathematics, allows us to separate the useful roots of the counted polynomial equations from those which are unneeded. Furthermore, we adress the problem of redundant event computation which consumes an indispensable amount of the runtime (almost 50% of the events is computed but not executed). Despite the deficiency in the fields of speed and stability of our current implementation of the algorithm, we show that a large performance enhancement is theoreticaly possible by recognizing and not computing the redundant topologic events.

CR Categories: G.1.5 [Numerical Analysis]: Roots of Nonlinear Equations—Polynomials, methods for; I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling—Geometric algorithms, languages, and systems

Keywords: Kinetic Delaunay Triangulation, Polynomial, Sturm Sequence

1 Introduction

Delaunay triangulation (DT) constructed over a set of time-dependent data (also called kinetic DT) represents a multi purpose data structure that may be used in various fields of computer science. The most commonly discussed use of kinetic DT is collision detection in environments with moving obstacles - see [Gavrilova et al. 1996] - because certain features of DT ensure that there will be an edge between each two points that are about to collide and thus we only have to test for collision the pairs of points which are connected by an edge of the DT. Another example of a similar application of kinetic DT is path planning in similar kinds of environment. In this case the triangulation is transformed to its dual Voronoi diagram which is then used to find the most suitable path. An example of such an application is for instance the marine vessel navigation as described in [Gold and Condal 1995]. We have also used the kinetic DT as a base data structure for video representation. In this application the movement of the points in the triangulation describes the correspondency between two consecutive frames of the video sequence and the triangulation is reinitialized upon reaching a keyframe. More information on this topic may be found in [Puncman 2008; Vomáčka 2008c]. Another example of possible use of kinetic DT are virtual reality environments. The kinetic DT may then serve for the purpose of communication with the user who then determines the movement of the kinetic data by the actions he or she takes inside the virtual world.

This paper focuses on the problem of maintaining a kinetic Delaunay triangulation with points that move along linear trajectories with no acceleration. We use the continous movement approach with the topologic events being stored in a priority queue. Furthermore, we focus on the analysis of the computation of the times of these events and propose some enhancement for the method we use for the computation.

This paper is organized in the following fashion: Section 2 describes the known and most often used approaches for solving the given problem. Section 3 provides the necessary definitions. Section 4 focuses on the computation and handling of the topologic events. Section 5 introduces the Sturm sequences of polynomials to the reader and shows how they are used in the process of solving the polynomials. Section 6 describes the results we achieved with our implementation of the described algorithm. Section 7 summarizes the whole paper. A preliminary version of the method has already been presented in [Vomáčka 2008a]. The performance and stability of the method has been strongly improved since the ver-

[*] e-mail: tvomacka@kiv.zcu.cz

[†] e-mail: kolinger@kiv.zcu.cz ; Work has been supported by the Ministry of Education, Youth and Sports of the Czech Republic under the research project LC-06008 (Centre for Computer Graphics).

sion presented there. The enhancements in both of those fields are in detail presented in [Vomáčka 2008b; Vomáčka 2008c] and this paper describes only their most essential parts. However, additional possibilities of further improving the stability and performance are given in this work.

2 State of the Art

Two main approaches for managing DT of moving data have been developed. The first of them separates the movement into discrete time instants and then utilizes repeated removal of the moving points from the triangulation followed by reinserting these points back into the triangulation at new coordinates. Such a triangulation is then often referred to as a fully dynamic DT. This approach has some obvious disadvantages resulting from the discrete understanding of the time of the moving objects. The possibly most important one of them is the fact that this approach may not be used in quite a large area of applications (such as collision detection) due to the risk of missing some important events that may occur between two consecutive discrete times. This situation may occur for instance in the case that some points in the triangulation are moving relatively fast and their new position is thus relatively far away from their original coordinates. On the other hand, this approach is extremely simple and requires no additional computation besides the triangulation construction (in this case, those algorithms which are online are considered to be the best ones - see [Vomáčka 2008b]) and point removal. It is also unaffected by the type of movement of the points, which may be extremely useful in certain types of applications. Algorithms for removing points from Delaunay triangulation are described for instance in [Devillers 1999; Mostafavi et al. 2003] and many others. A detailed review of the dynamic data structures including the use of this approach for movement simulation may be found in [Mostafavi et al. 2003]. Another example of this method is presented in [Thibault and Gold 2000].

The second approach is based on the idea that the topology of the triangulation changes only at certain time instants called topologic events - see [Gavrilova et al. 1996]. It has been shown that these events occur only when four points in the triangulation become cocircular as a result of their movement. Between two consecutive topological events, all the points in the triangulation may move without any restrictions (except the type of their trajectory which has to be linear in the vast majority of the cases). The algorithms which utilize this approach have to solve two important problems - how to obtain the times of the topologic events and how to handle these events. In the case that the triangulation is constructed using Euclidean metrics and the points move along linear trajectories with no acceleration, the event times may be found by solving polynomial equations of degree up to four - see [Albers et al. 1998]. The events may then be stored in a priority queue with the priority being the time of their occurrence thus allowing us to update the state of the triangulation by popping events from the queue, processing them and eventually pushing new ones back into the queue as they are computed during the process. Another way to process the topologic events is to split the continuous time of the triangulation into very short intervals as described in [Ferrez 2001] and postpone the evaluation of the events during each interval to its end. This can, however, lead to some numerical inconveniences and in the case of collision detection even to missing a collision, but if the intervals are short enough, the triangulation state will be legal at the end of each interval. Another method that can be used for evaluating the movement of the points is described in [Schaller and Meyer-Hermann 2004] as a method for point removal. It is computationaly very simple but may only be used when only one point is moving at a time.

It is also common that the papers which use the above mentioned continuous movement approach (see above) only state that the polynomials have to be solved but give no information on what is the best way of solving them. These polynomials cannot be in practice solved analytically due to floating point arithmetics imprecision. On the other hand, from all the available methods for solving polynomials (or nonlinear equations in general), only few are suitable for the discussed problem because of their unnecessary complexity or their focus on finding all of the complex roots of the given polynomial which is both unnecessary and unwanted (introducing the complex numbers into this kind of computation may increase the degree of imprecision). Overview of these methods (which include, but are not limited to, Lehmer-Schur method, Bairstow method, Bernoulli's method and others) may be found for instance in [Pan 1997; Ralston 1965].

3 Definitions

3.1 Triangulation and Delaunay Triangulation

Triangulation $T(S)$ of a set of points S in the Euclidean plane is a set of edges E such that

- no two edges in E intersect in a point not in S,
- the edges in E divide the convex hull of S into triangles

Delaunay triangulation $DT(S)$ over a finite set S of n points in the Euclidean plane

$$S = \{P_1, P_2, ..., P_n\}$$

is the triangulation that fulfills the condition that no point $S_i \in S$ is inside any triangle in $DT(S)$ - see [Hjelle and Dæhlen 2006]. This property is sometimes also called the Delaunay criterion or the empty circumcircle criterion. Because this condition determines the Delaunay triangulation, it is thus crucial that it is preserved during the whole lifetime of the algorithm despite the point movement. To determine if a point and a triangle satisfy this criterion a matrix test is performed - it is often called the incircle test and is described further.

3.2 Incircle Test

Let us have a triangle $P_1P_2P_3$ and a point P_4, to determine whether the point lies inside, on or outside the circumcircle of the triangle we have to compute the determinant of the following matrix $\mathbf{I}$:

$$\mathbf{I} = \begin{bmatrix} x_1 & y_1 & x_1^2 + y_1^2 & 1 \\ x_2 & y_2 & x_2^2 + y_2^2 & 1 \\ x_3 & y_3 & x_3^2 + y_3^2 & 1 \\ x_4 & y_4 & x_4^2 + y_4^2 & 1 \end{bmatrix} \tag{1}$$

where $P_i = [x_i, y_i]$ are the coordinates of the point P_i in the Euclidean plane.

By computing the value of $\det \mathbf{I}$ we can determine the mutual position of the point against the circumcircle of the triangle. For instance, if the triangle $P_1P_2P_3$ is oriented counterclockwise and $\det \mathbf{I} > 0$ then point P_4 lies inside the circumcircle of the given triangle and value of $\det \mathbf{I} = 0$ always means that the four points are cocircular, despite the orientation of the triangle. Detailed information on the incircle test may be found in [de Berg et al. 1997]

3.3 Point Movement

Points $P_1, ..., P_n$ are moving at a constant velocity and their coordinates must be thus defined as linear functions of time:

$$P_i(t) = [x_i(t), y_i(t)] \quad (2)$$

$$x_i(t) = x_i^0 + \Delta x_i \cdot t \quad (3)$$

$$y_i(t) = y_i^0 + \Delta y_i \cdot t$$

where $t \geq 0$ is the current time of the triangulation (i.e., the time since the movement has started) and $P_i(0) = \left[x_i^0,\ y_i^0\right],\ x_i^0,\ y_i^0 \in \mathbb{R}$ is the initial position of point P_i, i.e., the position at which it was inserted into the triangulation and $v_i = [\Delta x_i, \Delta y_i]$ is the velocity vector of point P_i.

Additionally we require each of the points to have its initial position (the position for $t = 0$) to be inside a certain triangulation area - a rectangle in Euclidean plane defined as:

$$\mathbf{O} = \langle x_{min}, x_{max} \rangle \times \langle y_{min}, y_{max} \rangle \quad (4)$$

where $x_{min}, x_{max}, y_{min}, y_{max} \in \mathbb{R}$ are the boundaries of the triangulation area. No point may then leave area **O** and if it is about to move outside of the given bounds, a collision will occur and change the velocity of the point in such a fashion to keep it inside the boundaries.

3.4 Topologic Event

As shown in [Albers et al. 1998; Gavrilova et al. 1996] and others, if the triangulation contains at least one point with nonzero velocity vector, its structure will have to change in time in order to keep the Delaunay condition valid. The moving points may change their position freely until they reach such a position when the change is inevitable and topologic event occurs - see Fig. 2.

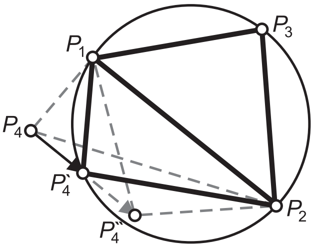


Figure 2: *Triggering of a topologic event.*

As shown in this figure, the topologic event occurs when four points become cocircular and it is thus determined by the time a point (point $P_4 \rightarrow P_4' \rightarrow P_4''$ here) enters a circumcircle of a triangle $P_1P_2P_3$. At this point the triangulation becomes non-Delaunay because the Delaunay condition is violated and the triangulation must be repaired by processing the topological event which is done by swapping the edge common to the two triangles in question - i.e., triangles $P_1P_2P_3$ and $P_1P_4'P_2$ here become $P_1P_4'P_3$ and $P_2P_3P_4'$ (illustration is given in Fig. 3, where point P_4 has moved inside a circumcircle of the triangle $P_1P_2P_3$. The resulting edge swap then changes the local topology as displayed in the figure). It is also important that the change in the topology of the triangulation is local, as shown in [Gavrilova et al. 1996], however more than one event may occur at the same time.

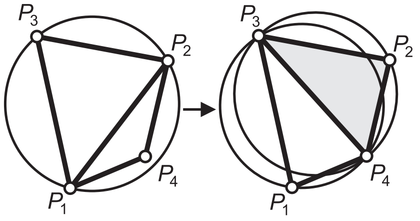


Figure 3: *Processing of a topologic event.*

4 Topologic Event Handling

4.1 Topologic Event Computation

In order to compute the time of a topologic event, we have to determine the time when four points that define two adjacent triangles become cocircular. This can be done by solving an equation created by establishing Eq. (3) into Eq. (1):

$$\det \mathbf{I}(t) = \det \begin{bmatrix} x_1(t) & y_1(t) & x_1^2(t) + y_1^2(t) & 1 \\ x_2(t) & y_2(t) & x_2^2(t) + y_2^2(t) & 1 \\ x_3(t) & y_3(t) & x_3^2(t) + y_3^2(t) & 1 \\ x_4(t) & y_4(t) & x_4^2(t) + y_4^2(t) & 1 \end{bmatrix} = 0 \quad (5)$$

where $x_i(t)$ and $y_i(t)$ are time dependent coordinates of the points in the triangulation as defined in Eq. (3). Because the coordinates of the moving points are linear functions of time (we consider them to move along linear trajectories with no acceleration), we can see that to solve Eq. (5) means to solve a polynomial equation of degree four or less depending on how many of the points in the two adjacent triangle configuration are actually moving.

4.2 Using the Priority Queue

As the points are inserted into the triangulation, the first future topology event is computed for each pair of adjacent triangle (if such exists) and these events are then placed into a priority queue. This process is often called the initialization step of the kinetic DT - see [Vomáčka 2008a]. From this queue the events are then popped in the order in which they occur and are executed. The process of executing a topologic event consists of swapping the common edge of the two triangles that defined the event as shown above. And because of the fact that two triangles are removed from the triangulation and are replaced by two new ones, new topologic events may occur and some of the previously computed events may become illegal. If any new events are computed, they are pushed into the priority queue and all of the now illegal events are removed from it without executing them. The *pop-execute-push* cycle is then repeated as needed in order to preserve the Delaunay condition of the triangulation and is called the iteration step.

4.3 Redundant Event Computation

It is vital to note that many of the computations of topologic events are redundant - some of the events will have to be computed more than once or will be computed but will never be processed. Let us consider the situation displayed in Fig. 4 and 5. In order to make the problem simpler to observe, the points in the figures that describe this problem are moved subsequently (the point P_4 remains static during the movement of P_5 and vice versa). From the definition of our approach, the points should move simultaneously. In that

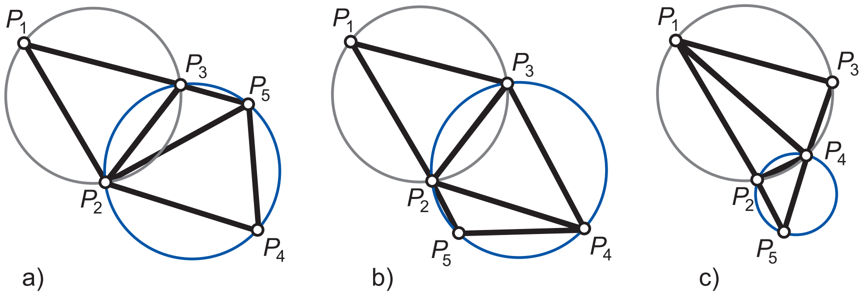


Figure 4: *Topologic changes in the vicinity of an upcoming event.*

case the speed of movement of P_4 would be much slower than the speed of P_5. The consecutive movement of points reffers to both the Figures 5 and 4.

As we may see, the topologic event for triangles $P_1P_2P_3$ and $P_2P_4P_3$ will occur at the time t_0. Another topologic event is scheduled for triangles $P_2P_4P_3$ and $P_3P_4P_5$ and it will occur at the time t_1. Even another topologic event will occur for the same pair of triangles at the time t_2 but it is not stored in the queue yet because we only push the first future topologic event for each triangle pair. Please note that events which are taking places at the time t_1 and t_2 are computed with respect to the movement of the points P_4 and P_5 and thus will occur at the circumcircles of the triangles that do not exist yet (their topology is correct but the position of the points will change due to the movement). In order to keep the figure as simple as possible, these events are marked on the currently existing circumcircle which will change its radius due to the movement of point P_4.

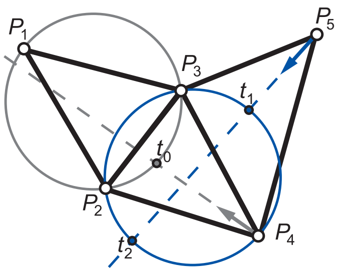


Figure 5: *Redundant computatation of topologic events.*

Let us then suppose that $t_1 < t_2 < t_0$. The events will occur in the following order: t_1, t_2, t_0. In this case the topology of the triangulation changes at t_1 as shown in Fig. 4a, making the event scheduled for triangles $P_1P_2P_3$ and $P_2P_4P_3$ at t_0 illegal and removing it from the queue. After that, the topology changes again at t_2 (Fig 4b), reverting partialy to the original state. This change enforces the computation of a topologic event for the triangle pair $P_1P_2P_3$ and $P_2P_4P_3$ which results in the topology event at the time t_0 which has been already computed and has been discarded. The situation can be even worse when multiple points leave and enter the vicinity of a triangle pair similar to the one displayed in Fig. 5.

In the case that $t_1 < t_0 < t_2$ (the order of the occurence of these events is that the event at t_1 will occur as the first one, supposedly followed by the event at t_0 and then by the event at t_2). The situation is similar to the previous case - the first topologic event, scheduled at t_1 makes the event at t_0 illegal and it is then removed from the queue. But due to the fact that $t_0 < t_2$, this exact topology event will never occur because the triangulation does not return to its original topologic state (or if it does so it is not soon enough for the original event to be replaced by its exact copy as in the previous case).

4.4 Singular Cases

As mentioned before, the times of topologic event are obtained by computing a polynomial equation of degree up to four in the form described in Eq. (5). The events (if some exist) are then denoted by the roots of this equation. The location and multiplicity of the roots may then give us some vital information whether and how should they be processed. The most obvious use of this information is the fact that we generally want to use only those roots that take place in the future and of them usualy only the first ones. So if the current time of the triangulation is equal to $t = t_{curr}$ and we obtain roots of t_1, t_2, t_3, t_4 as a result of solving the equation in the form of the one shown in Eq. (5) then if $t_1 < t_2 < t_{curr} < t_3 < t_4$, the roots t_1 and t_2 are obviously out of our focus and we only push into the queue the event that takes place at t_3. This principle has, however, some less obvious but maybe even more important use. It allows us to recognize singularities caused by the movement of the point which must be solved in a special fashion. One example of such a singularity is a tangential movement of a point towards a circumcircle of an adjacent triangle. An illustration of such a case is given in Fig. 6. Other types of singularities are described in [Vomáčka 2008a; Vomáčka 2008b].

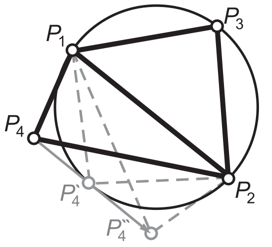


Figure 6: *Tangential movement of a point against a circumcircle.*

What we see in Fig 6 is that the point $P_4 \rightarrow P_4' \rightarrow P_4''$ moves tangentially towards the circumcircle of the triangle $P_1P_2P_3$. At the time it reaches the position marked as P_4', both of the available triangular configurations are legal (i.e., the edge swap may occur, but it is not necessary). However, for each other position of P_4, only the displayed triangle configuration is legal. This feature is a result of the fact that the root obtained by solving the appropriate equation is of double multiplicity and thus two edge swaps should occur. Knowing that two edge swaps result in exactly the same configuration as was the one we started with, we may as well skip both of them. The same rule applies to each root of even multiplicity - see [Vomáčka 2008b].

5 Sturm Sequences of Polynomials

5.1 Mathematical Requirements

Because of very high expected amount of computed polynomial equations (an equation will be computed for each pair of adjacent triangles and one equation will be computed per each executed topologic event), it would be useful if we had a mathematical tool that would help us to compute only such roots of these equations that we really need and can use. The Sturm sequences of polynomials, which are defined later in this section, allow us to do this by splitting the domain of the solved polynomials into suitable intervals, each of which contains exactly one root (ignoring the multiplicities of these roots). This feature allows us to take full advantage of the fact described earlier in Section 4.4 that we are only interested in the future topologic events (and usually even only in the nearest one of them). Combined together with the ability to separate the roots of a polynomial, we are able to run the time-consuming iterative computations only on a relatively small portion of the interval we would otherwise have to search.

The Sturm sequences of polynomials also enable us to detect (and possibly avoid) the singular cases mentioned above. These singularities are recognized by even multiplicities of the roots which denote the topologic events in question or degenerated cases of the polynomials (for instance $0 \cdot t = 0$). One of the most important features in the singular case detection is the fact that they may be easily detected immediately after the Sturm sequence is created. The detection is possible because the roots of the last term of the newly constructed Sturm sequence directly corresponds with the multiple roots of the original polynomial (see later).

5.2 Definition

As defined in [Ralston 1965], the sequence of polynomials

$$f_1(t), f_2(t), ..., f_m(t)$$

will be Sturm sequence (or Sturm chain) at interval $\langle a, b\rangle$ (a and b may be infinite), if:

1. $f_m(t)$ is nonzero at the whole interval $\langle a, b\rangle$
2. The two adjacent polynomials to the polynomial $f_k(t), k = 2, ..., m-1$ are nonzero at zero points of this polynomial and have the opposite signs there, thus:

   $\forall t \in \mathbb{R} : f_k(t) = 0 \Rightarrow f_{k-1}(t)f_{k+1}(t) < 0, k = 2, ..., m-1$

5.3 Construction

Sturm sequence of a polynomial $f(t)$ may be constructed (as proved in [Ralston 1965]):

$$\begin{array}{rcl} f_1(t) & = & f(t) \\ f_2(t) & = & f'(t) \\ f_{j-1}(t) & = & q_{j-1}(t)f_j(t) - f_{j+1}(t), j = 2, ..., m-1 \\ f_{m-1}(t) & = & q_{m-1}(t)f_m(t) \end{array} \tag{6}$$

In these relations, $q_{j-1}(t)$ is the quotient and $f_{j+1}(t)$ is the negation of the remainder of division of the polynomial $f_{j-1}(t)$ by the polynomial $f_j(t)$. $\{f_i(t)\}$ is thus a sequence of polynomials of a decreasing degree. The first term of the sequence is the input polynomial, the second term is its derivate and each of the following terms $f_i(t)$ is obtained by computing the remainder of the division $\frac{f_{i-1}}{f_{i-2}}$ and changing the sign of this remainder.

These facts may become easier to observe for the reader, if we rewrite the third equation from Eqs. (6) to the following form:

$$\frac{f_{j-1}(t)}{f_j(t)} = q_{j-1}(t) + (-1) \cdot \frac{f_{j+1}(t)}{f_j(t)} \tag{7}$$

What we see in Eq. (7) is a division of two polynomials, with $f_{j-1}(t)$ being the numerator and $f_j(t)$ being the denominator of the division. $q_{j-1}(t)$ then denotes the quotient (which is unused for the creation of the Sturm sequence) and $f_{j+1}(t)$ is the negation of the remainder of the division (the multiplication of $f_{j+1}(t)$ by the constant -1 is necessary to make the relation mathematically correct).

5.4 Important Features and Generalization

Counting the Roots Let us define a function $V(t)$ as the count of the number sign changes in the Sturm sequence as in Eqs. (6) (ignoring all zeros). This function may then be used to count the number of distinct real roots of $f(t)$ on any interval $\langle a, b\rangle$:

$$r_{\langle a,b\rangle} = V(a) - V(b) \tag{8}$$

where $a, b \in \mathbb{R}$ or either of a, b may be infinite. As proved in [Ralston 1965], Eq. (8) remains valid even if a or b are the roots of $f(t)$.

Root Multiplicity The last term of the Sturm sequence as in Eqs. (6) may be used to distinguish and compute the values of the multiple roots of $f(t)$. As proved in [Ralston 1965], all the multiple roots of $f(t)$ with multiplicities decreased by one are the roots of $f_m(t)$, which does not have any other roots. Together with the fundamental theorem of algebra, this statement may be extended to various useful conclusions. For instance if $f_m(t)$ is of an odd degree, then $f(t)$ has at least one multiple root, etc.

Note that, if the initial polynomial has some multiple roots, the created sequence is no further a Sturm sequence as defined in Section 5.2, because the second required condition is not met. In this case, the sequence is called a generalized Sturm sequence and has all the aforementioned features. The generalized Sturm sequence is formally defined as an extension of Sturm sequence $\{f_i(t)\}$ by multiplying all of its terms by any polynomial $p(t)$, thus gaining a sequence in the form of $\{p(t) \cdot f_i(t)\}$. If a Sturm sequence is mentioned anywhere in the following text, a generalized Sturm sequence is meant.

5.5 Solving the Polynomials Using Sturm Sequences

The process of solving a polynomial (in order to obtain the times of topologic events) consists of several steps (we only solve polynomials of degree greater than two because the linear and quadratic equations may be solved analytically with sufficient precission.). At first, a Sturm sequence is constructed for the given polynomial. We then attempt to solve its last term in order to find the multiple roots of the original polynomial. If there are any of them, we divide the original polynomial by $(t - t_i)^m$, where t_i is the multiple root and m is its multiplicity and we proceed with the result of this division to find the remaining roots. If there are none multiple roots, we solve the derivate of the original polynomial in order to estimate the intervals which contain its roots and then use a suitable numerical method to locate them (for instance the Newton method). The reason why this process is possible is that the number and multiplicities of roots of a real polynomial are determined by the fundamental theorem of algebra - see [Weisstein 2004]. This theorem determines that the number of complex roots of a real polynomial is either even or zero and thus allows us to limit the possibile combinations of the number and multiplicities of the roots of a polynomial to very small number which can further be reduced to one available combination by solving the derivate polynomial of the original polynomial equation.

The whole procedure of solving a polynomial of the third degree is sumarized in Algorithm 1, solving a polynomial of higher degrees is very similar to the described one. Additional details on this method may be found in [Vomáčka 2008b].

Algorithm 1: Sturm3 Algorithm

```
Input:
  • $p(t) = \sum_{i=0}^{3} a_i \cdot t^i = 0$ - a polynomial of the third degree
Output:
  • A sequence $\{t_i\}_{i=1}^{r}$ of the real roots of $p(t) = 0$, $r \leq 3$.
  • Or an empty sequence, if no real roots exist.
Auxiliary:
  • Sturm sequence $f_1(t), ..., f_m(t)$ of the polynomial $p(t)$ - see
    Eqs. (6), note that $f_1(t) = p(t)$.
  • $R_m = \{r_m^i\}_{i=1}^{r_{mult}}$ - a sequence of all the multiple roots of
    $p(t)$. Each multiple root $r_m^i$ is contained $m_i - 1$ times,
    where $m_i$ is its multiplicity.
// Create the Sturm sequence
$f_1(t), ..., f_m(t) \leftarrow$ Sturm sequence of $p(t) = f_1(t)$
$r \leftarrow (V(-\infty) - V(\infty))$// See Eq. (8)
if $r = 0$ then
    // $p(t)$ has no real roots
    // This situation may not occur for
       the polynomials of the third
       degree, but is possible for the
       polynomials of even degrees.
    Return an empty sequence $\{\}$ of roots.
end
// Obtain the multiple roots of $p(t)$
$R_m \leftarrow$ sequence of $r_{mult}$ roots of $f_m(t)$
if $\|R_m\| = 2$ then
    Return $\{r_{m1}, r_{m1}, r_{m1}\}$// One triple root
else if $\|R_m\| = 1$ then
    // $p(t)$ has a double and a single root
    $r_s \leftarrow$ the only single root of $\frac{p(t)}{(t-r_{m1})^2} = 0$
    Return $\{r_{m1}, r_{m1}, r_s\}$// A double and a
       single root
else
    // No multiple roots, solve $p(t)$, using
       a suitable numerical method
    Return $\{r_i\}_{i=1}^{r}$ ... sequence of $r \in \{1, 3\}$ distinct roots.
end
```

6 Results

6.1 Theoretical Results

In order to examine the theoretical expectations, we have implemented the algorithm described in this paper and run several tests. The most important of their results are presented in this section. All the tests described here were performed on random sets of points generated by using the following approach: 100 random points were placed in a bounding area 100×100 units in size. These points each had 1 unit in diameter safety disc (two points collide if their become at least as close as is the sum of the radii of their safety discs). Then certain percentage of the points were assigned a random velocity vector with each of its component being a random number from $\langle -5; 5 \rangle$ interval. This configuration was then observed for 10 seconds for different percentages of the moving points.

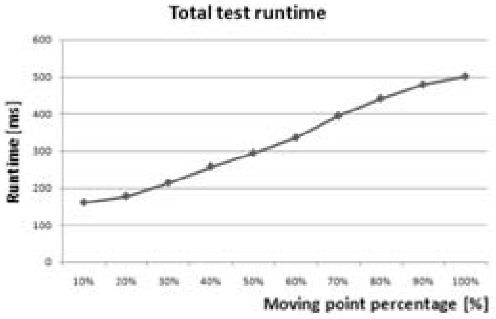


Figure 7: *Total runtime consumed by the test.*

Figure 7 shows the total runtime consumed by the tests as described above. According to the measured values, we may assume that the consumed runtime has an upper bound of $O(n)$ and a lower bound of $O(\log n)$.

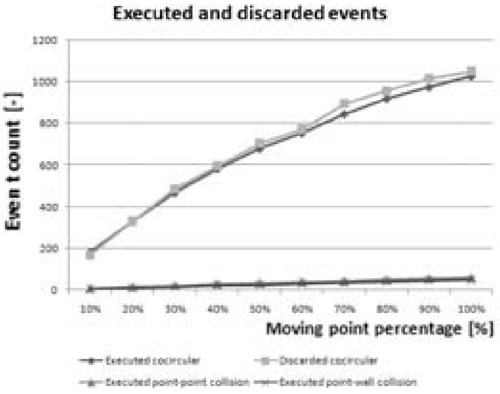


Figure 8: *Numbers of executed and discarded events of various types.*

In Fig. 8 we can see that very large number of the events is computed but not executed (these events are marked as discarded). Let us now ignore the collision type events which are unimportant because they only play a minor role in the total runtime consumption (they are caused by the collisions of points with walls of the bounding area - see Section 3 and by the collisions of the pairs of points). Assuming from this graph, we may state that the redundant computation of topologic events covers nearly as much as 50% of the runtime needed to compute the topologic events. Furthermore, we may assume that there is an upper bound of $O(n)$ and a lower of $O(\log n)$ on both the number of executed and discarded topologic events, which corresponds with the results presented in [Albers et al. 1998], where a naive estimation of the upper bound on the number of the events is given as $O(n^{d+2})$ and its linear improvement is presented.

6.2 Performance in Other Applications

As mentioned in Section 1, we have used our implementation of the described algorithm as a tool for video representation. Utilizing several techniques for stability improvement and overall performance enhancement we were able to successfuly use the kinetic DT as a basic data structure for video representation. A screenshot of the demonstration application is shown in Fig. 9 and additional details on this topic may be found in [Puncman 2008].

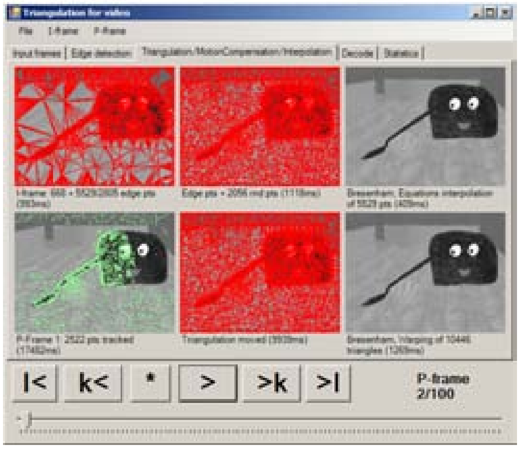

Triangulation for video
File
I-frame
P-frame
Input frames
Edge detection
Triangulation/MotionCompensation/Interpolation
Decode
Statistics
I-frame: 668 + 5529/2805 edge pts (393ms)
Edge pts + 2056 rnd pts (1118ms)
Bresenham, Equations interpolation of 5529 pts (409ms)
P-Frame 1: 2522 pts tracked (17482ms)
Triangulation moved (3939ms)
Bresenham, Warping of 10446 triangles (1269ms)
|<
k<
*
>
>k
>|
P-frame
2/100

Figure 9: *A screenshots of the video application.*

6.3 2.5D Wave Simulation

For demonstration purposes, we have created a simulation of a 2.5D wave. This simple application consists of a grid of points. Each point in this grid has slightly altered its $y-$coordinate by adding a small random number. One row of these points is then assigned a velocity vector in the form of $[0, y]$. If the moving row collides with the boundary of the triangulation area, it is deflected backwards without any loss of speed. If two points collide, then they switch their velocity vectors, meaning that the previously moving point is now static and vice versa. The points are then assigned a z coordinate by using an equation of a Gaussian curve with its peak being at the coordinates of the moving row of points. This creates the illusion of a wave as shown in Fig. 1.

Animation Download

Some example animations, including the output of the video representation application and the 2.5D wave simulation may be downloaded from:

`http://home.zcu.cz/%7Etvomacka/animations/`

7 Conclusion and Future Work

The presented method for maintaining the kinetic DT uses the Sturm sequences of polynomials combined with the fundamental theorem of algebra in order to estimate the location and multiplicities of the roots of the solved equations and thus is able to discover both the usable times of topologic events and the times which we cannot use. Even although the current version of the application uses only a certain portion of the presented possibilities, we hope that in the near future we will be able to utilize the knowledge in the fields of redundant topological event computation and others in order to vastly improve its performance.

The performed tests show us that even the existing version is usable in certain application, such as the video representation and for demonstration purposes (the simulation of the 2.5D wave). Future improvement capabilities lie in both of these areas and even include extension to higher dimensions, which is possible without any changes in the currently used mathematical apparatus (the described relations may be very easily extended to 3D). On the other hand, capabilities in the field of generalization of the type of movement is nearly impossible because the equations remain usable only for very narrow set of types of movement. This set includes polynomial trajectories but may not be extended to general movement described by nonlinear equations.

Acknowledgements

I would like to thank to my colleague *Petr Puncman* for providing a testing application and many helpful ideas and suggestions, especially in the fields of improving the stability and overall performance of our application.

References

ALBERS, G., GUIBAS, L. J., MITCHELL, J. S. B., AND ROOS, T. 1998. Voronoi diagrams of moving points. *International Journal of Computational Geometry and Applications 8*, 3, 365–380.

DE BERG, M., VAN KREVELD, M., OVERMARS, M., AND SCHWARZKOPF, O. 1997. *Computational geometry: algorithms and applications*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

DEVILLERS, O. 1999. On deletion in delaunay triangulations. In *Symposium on Computational Geometry*, 181–188.

FERREZ, J.-A. 2001. *Dynamic Triangulations for Efficient 3D Simulation of Granular Materials*. PhD thesis, cole Polytechnique Fdrale De Lausanne.

GAVRILOVA, M., ROKNE, J., AND GAVRILOV, D. 1996. Dynamic collision detection in computational geometry. In *12th European Workshop on Computational Geometry*, 103–106.

GOLD, C. M., AND CONDAL, A. R. 1995. A spatial data structure integrating GIS and simulation in a marine environment. *Marine Geodesy 18*, 213–228.

HJELLE, O., AND DÆHLEN, M. 2006. *Triangulations and Applications*. Berlin Heidelberg: Springer.

MOSTAFAVI, M. A., GOLD, C., AND DAKOWICZ, M. 2003. Delete and insert operations in Voronoi/Delaunay methods and applications. *Comput. Geosci. 29*, 4, 523–530.

PAN, V. Y. 1997. Solving a polynomial equation: Some history and recent progress. *SIAM Review 39*, 2 (June), 187–220.

PUNCMAN, P. 2008. *Použití triangulací pro reprezentaci videa, Diplomová práce*. University of West Bohemia, Pilsen, Czech Republic.

RALSTON, A. 1965. *A First Course in Numerical Analysis*. McGraw-Hill, Inc.: New York.

SCHALLER, G., AND MEYER-HERMANN, M. 2004. Kinetic and dynamic delaunay tetrahedralizations in three dimensions. *Computer Physics Communications 162*, 9.

THIBAULT, D., AND GOLD, C. M. 2000. Terrain reconstruction from contours by skeleton construction. *Geoinformatica 4*, 4, 349–373.

VOMÁČKA, T. 2008. Delaunay triangulation of moving points. In *Proceedings of the 12th Central European Seminar on Computer Graphics*, 67–74.

VOMÁČKA, T. 2008. *Delaunay Triangulation of Moving Points in a Plane, Diploma Thesis*. University of West Bohemia, Pilsen, Czech Republic.

VOMÁČKA, T. 2008. Use of delaunay triangulation of moving points as a data structure for video representation. Tech. rep., University of West Bohemia, Pilsen, Czech Republic.

WEISSTEIN, E. W., 2004. Fundamental theorem of algebra. From MathWorld - A Wolfram Web Resource. http://mathworld.wolfram.com/QuarticEquation.html.

Robust Distance-Based Watermarking for Digital Video

A. S. Abdul-Ahad,[*] Maria Lindén,[†] Thomas Larsson[‡]
School of Innovation, Design and Engineering
Mälardalen University
Västeras, Sweden

Waleed A. Mahmoud[§]
Department of Electrical Engineering
Al-Isra Private University
Amman, Jordan

Abstract

In this paper, a mechanism of a distance-based algorithm is highlighted and tested to invisibly watermark a digital video (cover object) using an iconic image (watermark object). The algorithm is based on the distances among the addresses of values of the cover object. These distances are used to make the embedding. The order of manipulating these distances are specified by the values of the watermark data which is dealt with serially. The algorithm achieves a self encryption key. Each watermark object has its unique pattern of distances at different possible lengths of distance bits. This enhances the complexity of sequential embedding. The blind and non-blind algorithm are tested using direct (spatial) and first level Two Dimensional Discrete Wavelet Transform (2D DWT) embeddings. The algorithm shows resisting and withstanding against the most important attacks. Some of these include lossy compression, frame averaging, frame swapping, and frame dropping.

CR Categories: I.4.9 [Computing Methodologies]: Image Processing and Computer Vision—Applications; E.4 [Coding and Information theory]

Keywords: watermarking, digital video, iconic image, direct embedding, distance bits, 2D DWT embedding, attacks

1 Introduction

Nowadays, the e-security is becoming an urgent requirement and among its impressive weapons are watermarking techniques. The forthcoming applications like wireless broadband access, Multimedia Messaging Service (MMS), video chat, mobile TV, HDTV content, Digital Video Broadcasting (DVB), minimal service like voice and data, and other streaming services for ”anytime-anywhere” are sooner or later at the door. This prosperous world of digital multimedia communication encounters undoubtedly an outburst of malicious interventions (i.e., copyright infringement like piracy, fraud, forgery, copying, etc). This situation necessitates the design of reliable and robust systems to protect and preserve the integrity and safe passage of any form of data. Invisible watermarking is one of the techniques that are available. Sooner or later this technique will be widely used in the Internet [Hui and Yeung 2003; Cox and Miller 2001; Maes et al. 2000].

The integrity system must include authorization, authentication, privacy, encryption and copyright protection policies. These policies can be entirely or partially watermarked in the original version of any multimedia application that need to be maintained against any abusers and hackers. Watermarking applications such as signature, trade mark, logo, biometrics like fingerprint, iris, voice and so on, are forming the main tools to achieve these policies. It is trustworthy to say that the watermarking technique gives something like immunity to the multimedia object when it is watermarked. And it achieves the highest degree of privacy. So, the technique ”watermarking” has promising future and will be popularly widespread and used throughout the world, especially, in the multimedia factories, in the courts, in the banks, in the hospitals, etc, where the privacy is the highest priority [Abdul-Ahad et al. 2008; Chae and Manjunath 1999].

2 Important properties of digital video

The term Video is a Latin term and means “I see”. Video is basically a three-dimensional array of color pixels. Two dimensions serve as spatial (horizontal and vertical) directions of the moving pictures, and one dimension represents the time domain. A data frame (video frame image) is the set of all pixels that correspond to a single time moment. Basically, a frame is the same as a still image [Hui and Yeung 2003; Cox and Miller 2001; Doërr and Dugelay 2004].

This paper deals specifically with digital video frames having the properties of 24 bits per pixel (8 bits for each primary color red, green, and blue). Each primary color has a fixed range of 256 hexadecimal values. These allowable 256 hexadecimal numbers of each primary color can be read with 256 floating point numbers. Generally, each value is either 0 or multiples of 0.0039216 [Abdul-Ahad et al. 2008].

Figure 1: *Original digital video1 and video2 (cover objects)*

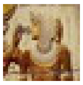


Figure 2: *Iconic image (watermark object)*

3 Direct embedding

Two cover objects, refereed to as digital video1 and video2, are used in this paper as shown in Figure 1. Both videos are of dimensions 240*320*3 with a frame rate of 30 fps and the encoding format is WMV (compressed video file format developed by Microsoft) . The watermark data may be an iconic image (the image may contain logo, signature, fingerprint, trade mark, serial number, and so on), text, or both. Figure 2 shows an iconic image with dimensions 32*32*3 (1024 pixels) to be used to watermark the cover objects.

The embedding process is implemented by decomposing the pixel values of the watermark object into packages of small numbers.

[*] amir_stephan@yahoo.co.uk
[†] maria.linden@mdh.se
[‡] thomas.larsson@mdh.se
[§] profwaleed54@yahoo.com

Thereby, for 2 bits distance, each pixel consists of 12-packages of small numbers. The total number of watermark data is 12288 and their four possible values are ranged from 1 to 4. For 4 bits distance, each pixel consists of 6-packages of small numbers. The total number of watermark data is 6144 and their sixteen possible values are ranged from 1 to 16. And for 8 bits distance, the total number of watermark data is 3072, ranged from 1 to 256 [Abdul-Ahad et al. 2008].

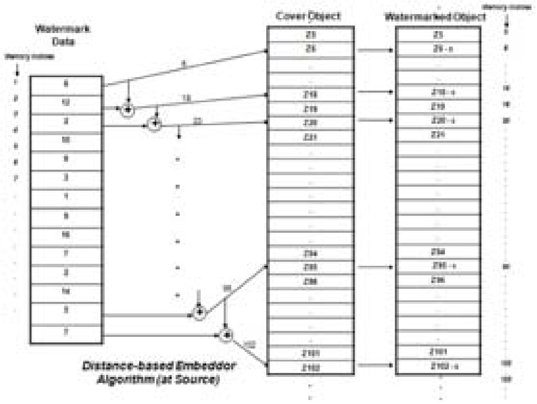


Figure 3: *Work of distance-based embeddor*

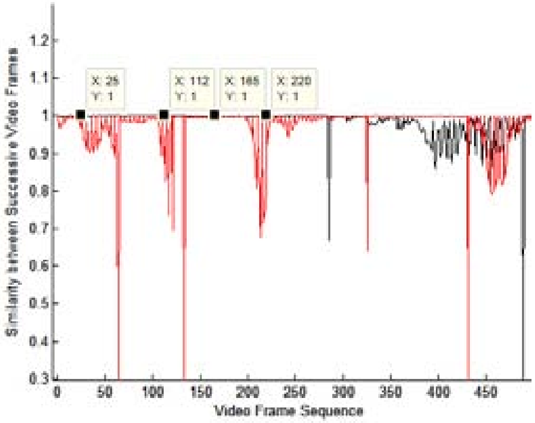


Figure 4: *Correlation between successive video frames. Red and black lines represent video1 and video2, respectively*

These values can be considered as addresses to locate the embedding positions in an uncompressed cover object (the embedding takes place before the WMV coder). The values are accumulated to get the next address. The difference in accumulated values of the watermark data can be thought of as distances among addresses of the cover object. This operation can be termed as Distance-Based Embeddor. The operation of this process is illustrated in Figure 3 and can be explained as follows: the value of the 1st watermark data points to the address of the 1st modified content. The 1st and 2nd watermark data are accumulated to get the 2nd address of the 2nd modified content and so forth. The distance between 1st address and 2nd address is equal to the value of 2nd watermark data. This accumulation goes on to the end of the last value of the watermark data. The contents of the positions which are addressed by the watermark data are modified with a factor of ε to obtain the watermarked object. This factor is selected to be the smallest value (zero is not selected) among the values of the color image of the cover object. In case of direct embedding, ε is selected to be equal to 0.0039216. This value has very little effect of distortion on the watermarked image. Thereby, the effect of uniquely distance-based deployment of watermark data in cover object helps in encryption. For example, for 2 bits distance-based embedding, the number of possible deployments of the watermark data over a cover object corresponds to the factorial of 12288 (i.e., 12288!). This reflects the complexity of the encryption per each iconic image. Besides, the possibility of embedding diversity at specific distance bits per frame (e.g., 2 bits distance, 4 bits distance and so on). It is worth mentioning here for non-blind watermarking, the embedding process can be anywhere in the cover object and doesn't matter how many times the watermark embed along the video. In case of blind watermarking, it is important to take in consideration the measurement of correlation for successive video frames to decide both the suitable place and how many times the watermark embed along the video (mostly successive video frames are highly correlated). As shown in Figure 4, the labeled video frames 25 and 112 for video1 and 165 and 220 for video2 are suitable for watermark embedding because of 100% correlation with successive video frames 26 and 113 for video1 and 166 and 221 for video2, respectively. This in turn, facilitates the blind watermark recovery. And at the same time this way of watermarking in many highly correlated video frames decreases the harming effect of frame dropping and frame swapping.

Peak Signal to Noise Ratio (PSNR) measures and mirrors the magnitude of distortion due to watermark data embedding in cover object. PSNR is apparently improved with an increase of distance bits. The reason is that for 2 bits distance, to embed 32*32 pixels, 12288 locations are required, while for 4 and 8 bits distances, only 6144 and 3072 locations are required, respectively. It is obvious that the number of modified values of watermarked object when using 8 bits distance is less than 4 and 2 bits distances. For 2 bits distance, the maximum and minimum distance between any consecutive addresses is 4 and 1 location(s), respectively, and the size of video frame must be at least 30498 bytes (sum the numbers of 2 bits 12288 values of iconic image) to totally embed an iconic image. For 4 bits distance, the maximum and minimum distance between any consecutive addresses is 16 and 1 location(s), respectively, and the size of video frame must be at least 46594 bytes (sum the numbers of 4 bits 6144 values of iconic image) to totally embed the same iconic image. And for 8 bits distance, the maximum and minimum distance between any consecutive addresses is 256 and 1 location(s), respectively, and the video frame size must be at least 401777 bytes (sum the numbers of 8-bits 3072 values of iconic image) to totally embed the same iconic image. Consequently, a different iconic image gives different lengths. It is clear that for the same watermark object, the 8 bits distance needs a larger cover object size to embed all watermark data in comparison to 4 and 2 bits distances. It is obvious that video file easily offers this possibility rather than audio and still image files, since, if needed, the deployment of the watermark data can be made over several video frames.

4 DWT embedding

The rapid progress of digital signal processing (DSP) techniques caused the sequential development of the electronic copyright (or proprietorship) protection. Due to this development, the DSP technique Discrete Wavelet Transform (DWT) becomes a vital and ready tool in achieving this requirement. It can be regarded as the most efficient transform dealing with multimedia applications since it provides a powerful and independent time-frequency (Multiresolution) representation.

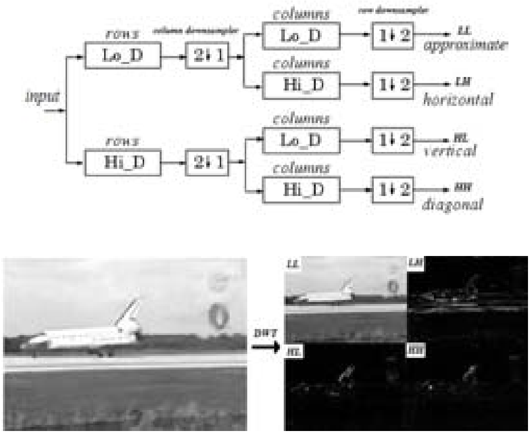


Figure 5: *Schematic diagram for the first level 2D DWT decompositions (top), and the result of the decompositions for the blue primary color (bottom)*

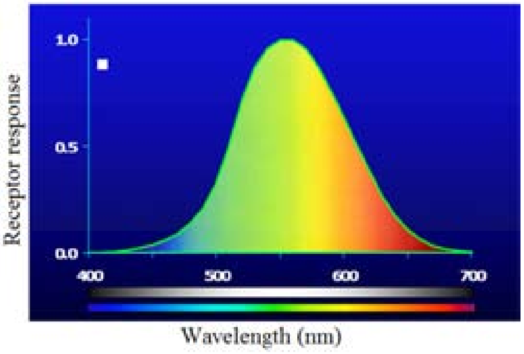


Figure 6: *Human eye sensitivity to spectral colors*

The least significant byte of a pixel's color (blue) of the video frame is decomposed into four sub-bands using 2D DWT technique as shown in Figure 5. As can be seen, any primary color channel can be transformed into four sub-bands, namely LL (approximate sub-band) where both horizontal and vertical directions have low frequencies, LH (horizontal sub-band) where the horizontal direction has low frequency and the vertical one has high frequency, HL (vertical sub-band) where the horizontal direction has high frequency and the vertical one has low frequency, and HH (diagonal sub-band) where both horizontal and vertical directions have high frequencies.

Each sub-band has dimension of 120*160 of 2D DWT coefficients. The blue color is preferred for watermark embedding, but if there is not enough space, the 2D DWT sub-bands of the red color is the next best option and the 2D DWT sub-bands of the green color is the last option. This choice is motivated by the fact that the human visual system (HVS) is subjectively less sensitive to the blue color and best sensitive to the green color as shown in Figure 6. The decomposition mechanism of the 2D DWT is very similar to the HVS mechanism. This means the higher resolution sub-bands LH, HL and HH are suitable regions to watermark embedding rather than the lower sub-band LL where the most primary color energy is concentrated. After the embedding, the 2D inverse DWT is applied to the four decompositions to get the watermarked object. Embedding watermark in these higher sub-bands increases the robustness of the embedding with no additional impact on video frame quality [Burrus et al. 1998; Kutte et al. 1998; Hunterlab 2001; Langelaar et al. 2000]. It is worthy to note that Daubechies wavelet function (db4) is used throughout this work.

5 Some models of attacking

Tables 1(a), 1(b), 2(a), 2(b), and 3 illustrate important results of attacking the watermarked object with effective and popular attacks [Doërr 2005; Alattar et al. 2003; K. Su 2001; Su et al. 2005]. The algorithm is tested by WMV compression, frame swapping, frame dropping, and frame averaging. In spite of all, signal to noise ratio (PSNR) readings are low as shown in Tables 1, 2, and 3. All the used attacks have a direct and visible effectiveness of distortion on the watermarked object. The proposed algorithm has the ability of high resilience of extracted watermark object regardless of what type of embedding technique is used. It is important to mention that all embeddings are implemented with 2 bits and 4 bits distances. The relevant results in the tables are very close to each other. This means that all embeddings are robust. However, according to the retrieving of the iconic image, it is apparent that the 1st level 2D DWT embedding is more withstandable than the direct embedding. Note that all the direct and 2D DWT embeddings used in Tables 1 and 2 are done at the same value of ε. This means that at amplified ε, Table 3 shows an improved version of the resilience of the extracted watermark shown in Table 1(a) using 4 bits distance-based direct embedding. Consequently, increasing ε increases the robustness of the watermark at the expense of the quality of the watermarked object.

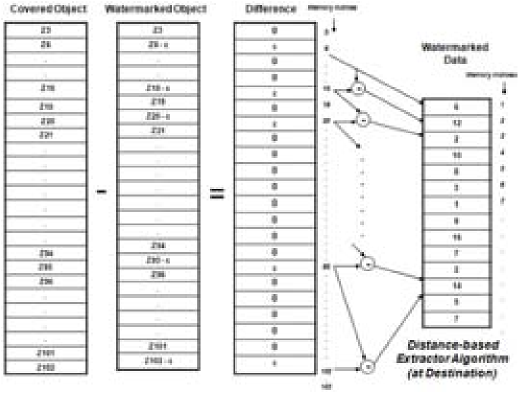


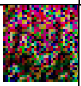
Figure 7: *Work of distance-based extractor*

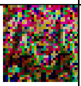
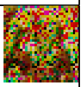
6 Recovering process

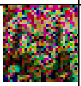
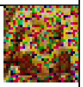
Figure 7 illustrates the work of the distance-based extractor. The watermark data is extracted from the difference between the cover and watermarked video frames in case of non-blind extraction, or between successive video frames in case of blind extraction. The difference contains numbers of zeros and εs in case of direct embedding or 1st level of 2D DWT embedding. The nonzero values indicate the addresses where the modification has taken place. The difference between any consecutive addresses of nonzero content is recovering the data of the iconic image. The recovered data must be

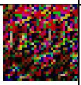
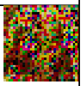
A) 4 bits distance-based direct embedding

Non-Blind Watermarking	
Video1	**Video2**

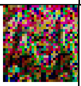
WMV Compression
PSNR = 41.1421
Corr = 1

	WMV Compression
	PSNR = 40.6394
	Corr = 1

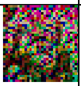
Frame swapping
PSNR = 39.6156
Corr = 1

	Frame swapping
	PSNR = 38.2422
	Corr = 0.9939
	

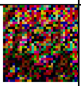
Frame dropping
PSNR = 37.8392
Corr = 0.9999

	Frame dropping
	PSNR = 25.6583
	Corr = 0.9801
	

Frame averaging
PSNR = 41.5083
Corr = 1

	Frame averaging
	PSNR = 36.2609
	Corr = 1
	

Blind Watermarking	
Video1	**Video2**

Frame swapping
PSNR = 41.7833
Corr = 1

	Frame swapping
	PSNR = 40.9564
	Corr = 0.9941

Frame dropping
PSNR = 38.7901
Corr = 1

	Frame dropping
	PSNR = 25.6615
	Corr = 0.9806

Frame averaging
PSNR = 42.7947
Corr = 1

	Frame averaging
	PSNR = 36.6966
	Corr = 0.9941

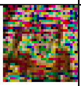
B) 2 bits distance-based direct embedding

Non-Blind Watermarking	
Video1	**Video2**

WMV Compression
PSNR = 37.8241
Corr = 1
 	
	WMV Compression
	PSNR = 37.4899
	Corr = 1

Frame swapping
PSNR = 37.1019
Corr = 0.9999
 	
	Frame swapping
	PSNR = 36.2532
	Corr = 1

Frame dropping
PSNR = 35.9362
Corr = 0.9999
 	
	Frame dropping
	PSNR = 25.5132
	Corr = 0.9976

Frame averaging
PSNR = 40.4198
Corr = 1

	Frame averaging
	PSNR = 33.9039
	Corr = 0.9997

Blind Watermarking	
Video1	**Video2**

Frame swapping
PSNR = 38.7691
Corr = 1
 	
	Frame swapping
	PSNR = 38.0842
	Corr = 1

Frame dropping
PSNR = 36.5731
Corr = 0.9999
 	
	Frame dropping
	PSNR = 25.5468
	Corr = 0.9975

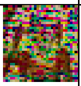
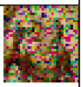
Frame averaging
PSNR = 40.9573
Corr = 1

	Frame averaging
	PSNR = 34.1585
	Corr = 0.9997
	

Table 1: *Attacks for distance-based direct embedding. Video1: compression (frame 25), swapping (frames 25, 26), dropping (frames 25, 27), and averaging (frames 23–26). Video2: compression (frame 165), swapping (frames 165, 166), dropping (frames 165, 167), and averaging (frames 163–166)*

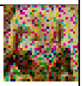
A) 4 bits distance-based first-level DWT embedding

Non-Blind Watermarking	
Video1	**Video2**

WMV Compression
PSNR = 38.2807
Corr = 1
 	
	WMV Compression
	PSNR = 38.1235
	Corr = 1

Frame swapping
PSNR = 37.1614
Corr = 0.9998
 	
	Frame swapping
	PSNR = 36.0716
	Corr = 1

Frame dropping
PSNR = 35.9163
Corr = 0.9998
 	
	Frame dropping
	PSNR = 25.5111
	Corr = 0.9975

Frame averaging
PSNR = 37.9260
Corr = 0.9998
 	
	Frame averaging
	PSNR = 24.4765
	Corr = 0.9974
	

Blind Watermarking	
Video1	**Video2**

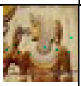
Frame swapping
PSNR = 37.1522
Corr = 0.9998
 	
	Frame swapping
	PSNR = 35.8883
	Corr = 1

Frame dropping
PSNR = 36.0045
Corr = 0.9998
 	
	Frame dropping
	PSNR = 25.5111
	Corr = 0.9975

Frame averaging
PSNR = 37.7667
Corr = 0.9997
 	
	Frame averaging
	PSNR = 26.6993
	Corr = 0.9978

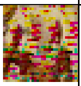
B) 2 bits distance-based first-level DWT embedding

Non-Blind Watermarking	
Video1	**Video2**

WMV Compression
PSNR = 37.9244
Corr = 1

	WMV Compression
	PSNR = 37.1998
	Corr = 1

Frame swapping
PSNR = 37.0914
Corr = 0.9999
 	
	Frame swapping
	PSNR = 36.0781
	Corr = 1

Frame dropping
PSNR = 35.9735
Corr = 0.9998
 	
	Frame dropping
	PSNR = 25.5779
	Corr = 0.9975

Frame averaging
PSNR = 39.9828
Corr = 1

	Frame averaging
	PSNR = 28.9937
	Corr = 0.9987

Blind Watermarking	
Video1	**Video2**

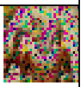
Frame swapping
PSNR = 38.0546
Corr = 0.9999
 	
	Frame swapping
	PSNR = 37.4842
	Corr = 1

Frame dropping
PSNR = 36.6066
Corr = 0.9998
 	
	Frame dropping
	PSNR = 25.8397
	Corr = 0.9978

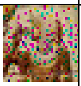
Frame averaging
PSNR = 39.7681
Corr = 1
 	
	Frame averaging
	PSNR = 29.4287
	Corr = 0.9991

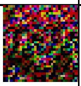
Table 2: *Attacks for distance-based DWT embedding. Video1: compression (frame 25), swapping (frames 25, 26), dropping (frames 25, 27), and averaging (frames 23–26). Video2: compression (frame 165), swapping (frames 165, 166), dropping (frames 165, 167), and averaging (frames 163–166)*

Non-Blind Watermarking	
Video1	**Video2**

WMV Compression
PSNR = 30.1508
Corr = 1
 	
	WMV Compression
	PSNR = 39.6619
	Corr = 1
	

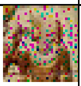
Frame swapping
PSNR = 38.9333
Corr = 1
 	
	Frame swapping
	PSNR = 37.8506
	Corr = 1

Frame dropping
PSNR = 37.3505
Corr = 0.9999

	Frame dropping
	PSNR = 25.6290
	Corr = 0.9976

Frame averaging
PSNR = 41.5257
Corr = 1

	Frame averaging
	PSNR = 36.2017
	Corr = 0.9999

Blind Watermarking	
Video1	**Video2**

Frame swapping
PSNR = 40.6039
Corr = 1
 	
	Frame swapping
	PSNR = 40.3088
	Corr = 1

Frame dropping
PSNR = 38.1280
Corr = 0.9999

	Frame dropping
	PSNR = 25.8901
	Corr = 0.9978

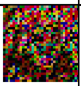
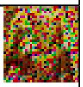
Frame averaging
PSNR = 41.8660
Corr = 1

	Frame averaging
	PSNR = 36.5163
	Corr = 0.9999
	

Table 3: *Attacks for distance-based direct embedding using amplified ε. Video1: compression (frame 25), swapping (frames 25, 26), dropping (frames 25, 27), and averaging (frames 23–26). Video2: compression (frame 165), swapping (frames 165, 166), dropping (frames 165, 167), and averaging (frames 163–166)*

12288 packages for 2 bits distance embedding and 6144 packages for 4 bits distance embedding.

The direct embedding algorithm is directly implemented on the content of the cover image and the amount of distortion depends upon ε and length of distance bits. While in distance-based 2D DWT embedding, in addition to ε and length of distance bits, the amount of distortion is affected by the order and level of wavelet selection.

7 Conclusion

The lossy compression constitutes the main obstacle and challenge to the technical watermarking but the algorithm was able to overcome it completely. Especially when using the technique DWT, this technique provides a kind of protection for the watermark data, particularly at higher resolution sub-bands, compared with direct embedding. Direct embedding is a pixel-wise process while DWT embedding is a coefficient-wise (indirect) process. Results written in the tables confirm this protection. In other words, the importance and benefits of watermarking technique will stimulate researchers to discover ways for lossless video compression.

Despite that PSNR is large, the resilience of the watermark image is vulnerable in front of the attacks particularly at 4 bits distance-based direct embedding, while at 2 bits distance-based direct embedding, the resilience is improved and robust. This is due to the wide deployment of 4 bits distance-based direct embedding over the frame, where the effect of lossy WMV compression is huge and devastating. This situation exists, but with little effect in case of 2D DWT embedding, where both 2 bits and 4 bits distance-based embeddings are robust.

Future research need to focus on decreasing the distortions and artifacts of the extracted watermark by extracting and embedding the most representative features of the watermark object. The goal would be to extend the proposed method to be resilient against other attacks such as geometric distortion of the images.

References

ABDUL-AHAD, A. S., CÜRÜKLÜ, B., AND MAHMOUD, W. A. 2008. Robust distance-based watermarking for digital image. In *Proceedings of the 2008 International Conference on Security and Management (SAM'08)*, 404–409.

ALATTAR, A., LIN, E., AND CELIK, M. 2003. Digital watermarking of low bit-rate advanced simple profile MPEG-4 compressed video. *IEEE Transactions on Circuits and Systems for Video Technology 13*, 8 (August), 787–800.

BURRUS, C. S., GOPINATH, R. A., AND GUO, H. 1998. *Introduction to Wavelets and Wavelet Transforms*. Prentice Hall.

CHAE, J., AND MANJUNATH, B. 1999. Data hiding in video. In *International Conference on Image Processing*, 311–315.

COX, I., AND MILLER, M. 2001. Electronic watermarking: the first 50 years. *IEEE Fourth Workshop on Multimedia Signal Processing*, 225–230.

DOËRR, G., AND DUGELAY, J.-L. 2004. Security pitfalls of frame-by-frame approaches to video watermarking. *IEEE Transactions on Signal Processing 52*, 10 (October), 2955–2964.

DOËRR, G. 2005. *Security Issue and Collusion Attacks in Video Watermarking*. PhD thesis, de lUniversit e de Nice-Sophia Antipolis.

HUI, S. Y., AND YEUNG, K. H. 2003. Challenges in the migration to 4G mobile systems. *IEEE Communications Magazine 41*, 12 (December), 54–59.

HUNTERLAB, 2001. The basics of color perception and measurement, version 1.4. www.hunterlab.com/pdf/color.pdf.

K. SU, D. KUNDUR, D. H. 2001. A content-dependent spatially localized video watermarked for resistance to collusion and interpolation attacks. In *IEEE International Conference on Image Processing*.

KUTTE, M., JORDAN, F., AND BOSSEN, F. 1998. Digital watermarking of color images using amplitude modulation. *Journal of Electronic Imaging 7*, 2 (April), 326–332.

LANGELAAR, G., SETYAWAN, I., AND LAGENDIJK, R. 2000. Watermarking digital image and video data. a state-of-the-art overview. *IEEE Signal Processing Magazine 17*, 5 (September), 20–46.

MAES, M., KALKER, T., LINNARTZ, J.-P., TALSTRA, J., DEPOVERE, F., AND HAITSMA, J. 2000. Digital watermarking for DVD video copy protection. *IEEE Signal Processing Magazine 17*, 5 (September), 47–57.

SU, K., KUNDUR, D., AND HATZINAKOS, D. 2005. Statistical invisibility for collusion-resistant digital video watermarking. *IEEE Transactions on Multimedia 7*, 1 (February), 43–51.

Interactive visualization of new electromagnetic quantities

Siavoush M. Mohammadi*
Swedish Institute of Space Physics, P. O. Box 537, SE-751 21 Uppsala, Sweden
Anders Hast†
UPPMAX, Uppsala University Box 337 SE-751 05 Uppsala Sweden
Lars K. S. Daldorff
Department of Physics and Astronomy, Uppsala University, P. O. Box 516, SE-751 20 Uppsala, Sweden
Martin Ericsson
UPPMAX, Uppsala University Box 337 SE-751 05 Uppsala Sweden
Jan E. S. Bergman
Swedish Institute of Space Physics, P. O. Box 537, SE-751 21 Uppsala, Sweden
Bo Thidé
Swedish Institute of Space Physics, P. O. Box 537, SE-751 21 Uppsala, Sweden

Abstract

Recent development in classical electrodynamics has demonstrated the usefulness of different rotational and topological modes in the electromagnetic fields (angular momentum, polarization, vorticity *etc.*). Unfortunately, the visualization tools available to illustrate these electrodynamic quantities have hitherto been inadequate. Therefore we have developed a VTK and Python based interactive visualization tool, with working name EMVT (ElectroMagnetic Visualization Tool), targeted at visualizing precisely these modes.

In the near future, EMVT will be further developed to visualize and control live antenna systems, where electromagnetic field data is instantly received, calculated, and visualized from an antenna or a system of antennas. It will then be possible to see how the antenna properties change through direct user interaction in real time.

1 Introduction

Ever since Poynting in 1909 [Poynting 1909] predicted that electromechanical rotations of a shaft must be carried out by electromagnetic (EM) radiation, the rotational modes of the EM fields have received more and more attention. Recent development has shown that there are no less than three different types of rotational degrees of freedom embedded in the fields [Bergman et al. 2008; Allen et al. 2003]. In analogy with a planetary body (for instance the Earth), an EM beam can rotate around its own axis (spin angular momentum), rotate around an external axis (orbital angular momentum) and nutate due to the tilt of the spin axis compared to the rotational plane around the axis (*cf.* Earth nutation). These rotational properties can be investigated by performing different multiplicative operations between the EM fields, time, and the position vector, resulting in different conserved physical quantities [Jackson 1998; Noether 1918; Bergman et al. 2008]. These conserved quantities are carried by the fields and, hence, can be used for information extraction. This makes them very important in physics [Thidé et al. 2007], astronomy [Harwit 2003], and communication. The relevant conserved electromagnetic quantities can be found in Table 1.

Standard visualization methods (see Fig. 2) are in essence limited to visualizing the power density, which is adequate for studies of the linear momentum of the fields. However, it does not yield satisfactory insight into the behavior of the rotations in the fields, which on the other hand are, for instance, clearly visible in Fig. 1.

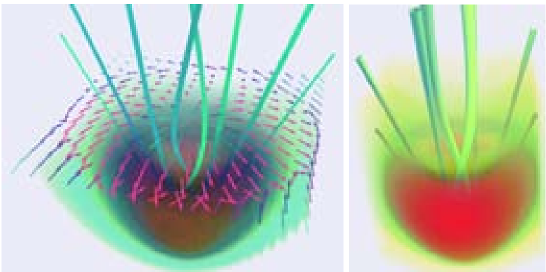


Figure 1: *The images show how the user can visualize the same data set differently by changing hue, saturation, and alphablend parameters, as well as using different visualization aids. The visualization aids used in the image to the left are volume rendering, stream lines with ribbon visualization, and a vectorial cut plane, whereas the image to the right only uses volume rendering and stream lines with tube visualization.*

2 Simulation setup

EM field rotations in the radio domain can be produced in a simple but powerful way with only a few antennas. Essentially, a polarized beam which also carries orbital angular momentum will have all of the three rotational modes embedded in the fields.

We have produced our raw data (EM vector fields) for the visualizations by using NEC2 [NEC2] and the geometry editor of 4NEC2 [Voors]. We placed ten identical, electrically short, crossed electric dipoles, 0.10 wavelengths (λ) above a perfect electric conduc-

Table 1: *Conserved Electromagnetic quantities, notation:* $u = \varepsilon_0(\mathbf{E}\cdot\mathbf{E}^* + c^2\mathbf{B}\cdot\mathbf{B}^*)/2$, $\mathbf{P} = \varepsilon_0\,\mathrm{Re}\left[\mathbf{E}\times\mathbf{B}^*\right]$, $\mathbf{V} = -\varepsilon_0\,\mathrm{Im}\left[\mathbf{E}\times\mathbf{E}^* + c^2\mathbf{B}\times\mathbf{B}^*\right]/2c$ *and* $w = 1\,\mathrm{Im}\left[\mathbf{E}\cdot\mathbf{B}^*\right]/Z_0$. *Abbrevations: AM = Angular Momentum, EM = Electromagnetic, Pol. = Polarization, Lin. mom. = Linear momentum,* ε_0 *= vacuum permittivity,* c *= speed of light,* Z_0 *= vacuum resistance.*

	Energies	Momenta	Angular Momenta
Usual	u	$\mathbf{P}$	$\mathbf{r}\times\mathbf{P}$
Phys. interp.	EM Energy	lin. mom.	AM
Pol. based	w	$\mathbf{V}$	$\mathbf{r}\times\mathbf{V}$
Phys. interp.	Pol. Energy	Pol. Vector	Spin orbital AM

*e-mail: simo@irfu.se
†e-mail:aht@cb.uu.se

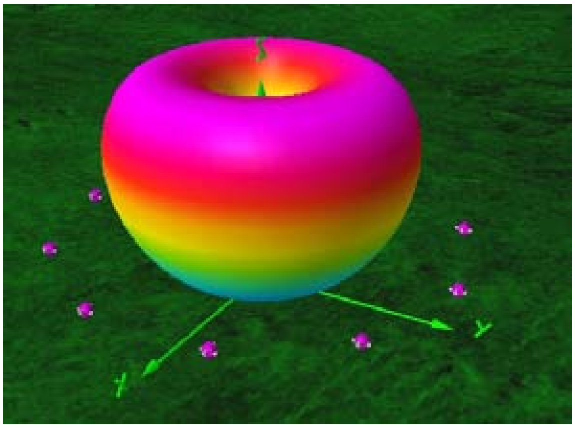


Figure 2: *A 3D visualization of the so called antenna diagram, corresponding to the structure of the electromagnetic power density produced by the ten element circular antenna array. Here, we see the general behavior of the array but there is no information about the rotations embedded in the fields. The image was created by 4NEC2.*

tor in a phased circular array; see Fig. 3. The largest dimension of the array is $D = \lambda$. The dipole elements are fed in such a way that a perfectly circularly polarized field, which also carries orbital angular momentum number proportional to unity, *i.e.*, $l = 1$, is created. The dimensions of the box, where the fields are calculated are: $-14\lambda < X < 14\lambda$, $-14\lambda < Y < 14\lambda$, and $0 < Z < 25\lambda$.

3 General visualization methods

3.1 VTK

We chose to use The Visualization ToolKit (VTK) [Kitware ; Kitware 2006] since it is an open source and freely available software system for computer graphics and visualization that supports a wide variety of visualization algorithms including scalar, vector and volumetric methods. Both the design and implementation of VTK has been strongly influenced by object oriented principles, which makes it relatively easy to use even though the library literally contains thousands of classes. However, the huge number of classes is probably a big threshold to overcome for those who wants to create their own visualizations for the first time and therefore some would for instance prefer Paraview [Squillacote 2008], which is an open source application designed to visualize large data sets. Nonetheless, we wanted to maintain full control over the whole visualization process and for our specialized application we wanted to be able to compute the new electrodynamic quantities within our application, instead of having a preprocessor stage. Furthermore, Python [Python ; Lindblad 2006] was used as the main programming language since it is object oriented and can handle data in an efficient way, *i.e.* it is easy to write a program using Python that reads the ascii output of NEC2 and converts it to the internal representation in VTK as well as it computes other quantities.

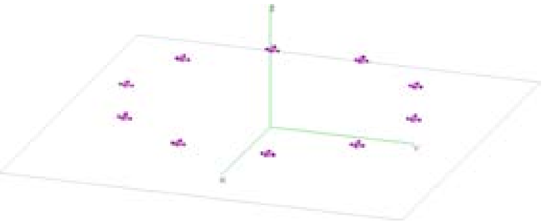


Figure 3: *Ten electrically short crossed electric dipoles in a circle, creating a field carrying a total OAM $l = 1$ and spin $s = 1$. The image is from the geometry editor of 4NEC2.*

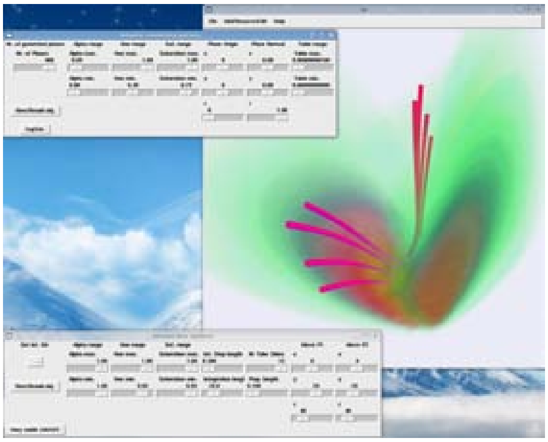


Figure 4: *A screen shot of EMVT. The user controls two objects which visualize the polarization and nutation of an electromagnetic field created by our simulation setup. All VTK objects here have either color transfer functions or so called vtkLookuptables, which control the hue, saturation, alpha blending, etc of the object. These properties, and others, can be changed interactively by the user.*

3.2 Methods used in EMVT

There are in principal five different types of visualization possible in EMVT:

1. **Volume rendering** – This type of rendering gives good insight into how the scalar quantities behave, where the colors are proportional to the scalar values. The intensity of vector fields (vector squared in each point) can also be visualized in this manner.
2. **Isosurfaces** – Scalar values or the amplitudes of a vector field can be illustrated by drawing a surface over all constant values of the considered quantity.
3. **Vector arrows** – Vector fields can be simply illustrated by vector arrows. The arrows can be normalized to be of the same length, where the coloring can be proportional to the length of the vectors or one of the scalar quantities.
4. **Cut planes** – Cut planes, can be used to cut both vector fields and scalar values. These can be oriented and placed as the user wishes.
5. **Flow lines** – Inspired by wind tunnel data visualizations, a “string” is let loose and guided by the vector field. In this way it is easy to see the flow of the field considered.

The visualization tools introduced in EMVT can be combined in whichever manner that suits the user and for any of the electromagnetic quantities. What is also an important feature of EMVT is that

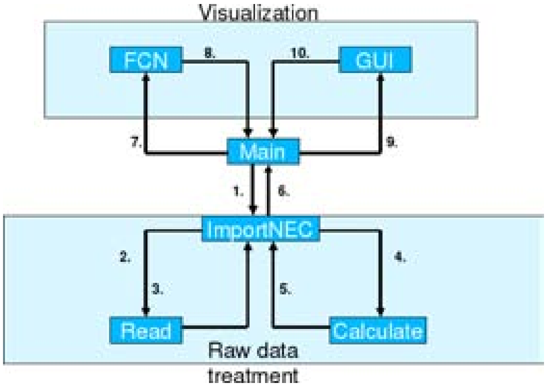


Figure 5: *This illustration shows the flow chart of EMVT. The program runs ten steps before an actual image is displayed.* ***1.*** *User input data files;* ***2.*** *Two strings, address to files;* ***3.*** *Dimension, NumberOfVectors, 4 × vtkDoubleArray, 2 × vtkPoints;* ***4.*** *nrVectors, 4 × vtkDoubleArray, 1 × vtkPoints;* ***5.*** *Treated data, 12 × vtkDoubleArray;* ***6.*** *Treated data, 12 × vtkDoubleArray, 1 × vtkPoints, nrVectors, dimensions;* ***7.*** *User chooses visualization tool and data to visualize, 1 × vtkDoubleArray;* ***8.*** *VTK objects and parameters;* ***9.*** *VTK objects and parameters; and* ***10.*** *VTKActor;*

it is possible to visualize as many different electromagnetic quantities at the same time in the same window as the user needs or wants. This is important since a general user of EMVT will have had first hand experience of the energy and linear momentum visualizations, for instance as in Fig. 2, but no experience of the visualization of the other EM quantities. When both the linear momentum and the angular momentum is visualized in the same image, the first visualization will provide a reference for the user, from which the angular momentum visualization can be put into context.

4 EMVT program structure

The main structure of EMVT is illustrated in Fig. 5. It consists of two different parts, the raw data processing, which reads data (“Read” class) and makes the necessary calculations (“Calculate” class) and the visualization classes (“FCN” and “GUI” classes). The raw data processing code is written in native Python whereas the visualization classes are written using VTKs class library. For each visualization type, *i.e.* volume rendering, cutplanes *etc.*, there exist two classes. The function class (FCN), which creates the necessary VTK objects and parameters, and the graphical user interface class (GUI), which in turn creates the GUI to control the VTK objects created by the FCN class. A screenshot of an EMVT run, where the user is interactively controlling two different objects to enhance the visualization of two electromagnetic quantities related to polarization, can be seen in Fig. 4. Other screenshots of the visualizations created by EMVT can be viewed in Fig. 1, 6 and 7. These images show different electromagnetic quantities visualized in different ways. Note the clear rotational patterns visible in Fig. 1 and 6, which visualizes the polarization, nutation and angular momentum of the field, whereas Fig. 7, which visualizes the Poynting flux (or linear momentum) with more traditional methods, gives no direct insight into the rotational structures. Furthermore, by comparing Fig. 1 and Fig. 8 the different rotational properties of the field can be clearly shown and thus the disparate nature of the spin rotations and orbital angular momentum rotations.

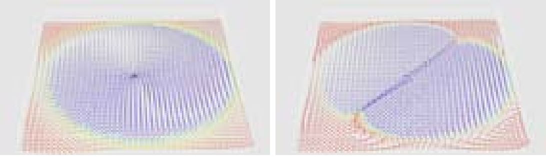


Figure 6: *These two images show how the nutation.* **r** × **V**. *of the electromagnetic field is affected when the polarization of the generated field is changed from perfectly circular (left image) to linearly (right image) polarization.*

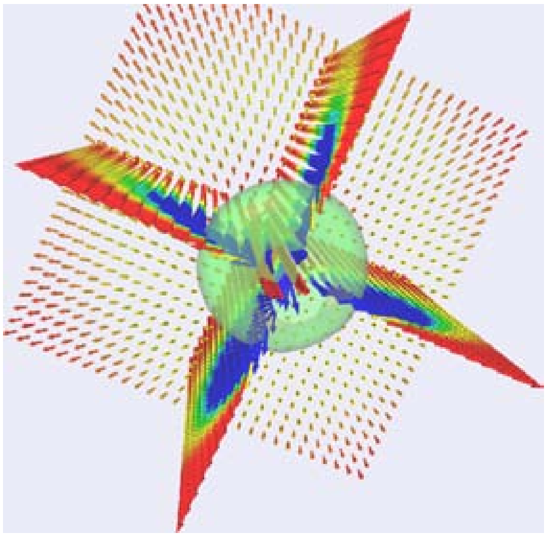


Figure 7: *This image visualizes the linear momentum (vectorial cut planes) and electromagnetic energy (isosurface).*

5 Discussion and future

Visualization of data is an indispensible tool for any physicist or scientist in general. By studying the graphs and visualizations created from the data the user can efficiently extract information of how the system works and, above all, get a feeling for it. Connections between different parameters and physical quantities can be enhanced, which otherwise would not be as obvious if one only studies mathematical expressions or the raw data sets. The need to be able to interactively tune and change visualization parameters as the alpha factor *etc.* is vital in order to enhance the structure one wants to see. Furthermore, in this application all of the visualized quantities, where calculated from the same dataset. This is an important feature since, generally, the output of any EM simulation program is typically the electric and magnetic vector fields. However, all of the physics of the system cannot be seen by only visualizing the fields. One needs to perform several calculations before the physical, observable, quantities of interest which reveal such behavior as rotations *etc.* are found.

5.1 Future

This visualization tool (EMVT) has the potential to be further developed and used on live antenna systems. A live antenna system can be anything from a single receiving antenna for an FM radio

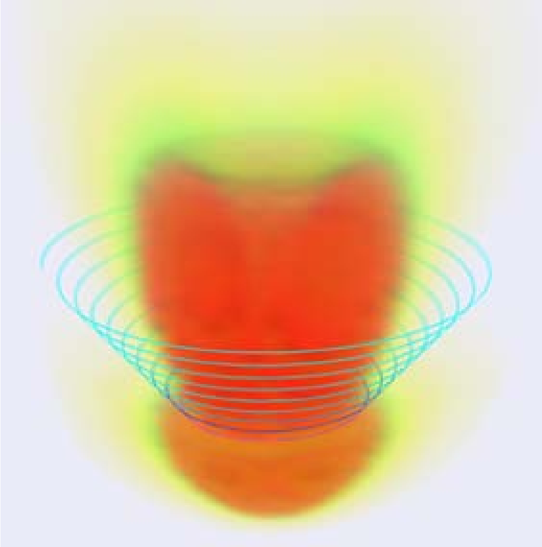


Figure 8: *This image shows the angular momentum intensity,* $|\mathbf{r} \times \mathbf{P}|^2$*, (volume rendering) and the angular momentum flow,* $\mathbf{r} \times \mathbf{P}$*, (stream lines) of the fields created by the described antenna system.*

to huge radio telescopes such as LOFAR [LOFAR]. By calculating the fields in a small volume around the receiving antenna or antennas from the currents induced on them, the user can visualize how the field and all its properties behave. Furthermore, if a controller is created for the receiving antenna then the user can instantly see how the antenna and different electromagnetic quantities change during the interaction with the application.

6 Acknowledgments

This work was financially supported by the Swedish National Space Board, the Swedish Research Council (VR). Uppsala Multidisciplinary Center for Advanced Computational Science (UPPMAX) is acknowledged for assistance concerning technical and implementational aspects in making the code run on the UPPMAX resources.

References

ABRAHAM, V. M. 1914. Der Drehimpuls des Lichtes. *Physik. Zeitschr. XV*, 914–918.

ALLEN, L., BARNETT, S. M., AND PADGETT, M. J. 2003. *Optical Angular Momentum.* IOP, Bristol, UK.

BERGMAN, J. E. S., MOHAMMADI, S. M., DALDORFF, L. K. S., THIDÉ, B., CAROZZI, T. D., KARLSSON, R. L., AND ERIKSSON, M. 2008. Conservation laws in generalized riemann-silberstein electrodynamics. *ArXiv 0803.2383*, v6.

HARWIT, M. 2003. Photon orbital angular momentum in astrophysics. *Astrophys. J. 597*, 2 (10 November), 1266–1270.

JACKSON, J. D. 1998. *Classical Electrodynamics*, 3 ed. Wiley, New York, ch. 7.

KITWARE. Visualization ToolKit (VTK). Web site. http://www.vtk.org/.

KITWARE, 2006. The Visualization Toolkit User's Guide.

LINDBLAD, E. 2006. *Programmering i Python.* Studentlitteratur, Lund.

LOFAR. Low Frequency Array. Web site. http://www.lofar.org.

NEC2. Numerical Electromagnetic Code, Version 2. Web site. http://www.nec2.org.

NOETHER, E. 1918. Invariante Variationsprobleme. *Nachr. Ges. Wiss. Göttingen 1*, 3, 235–257. English transl.: *Invariant variation problems*, Transp. Theor. Stat. Phys., **1**, 186–207 (1971).

POYNTING, J. H. 1909. The wave motion of a revolving shaft, and a suggestion as to the angular momentum in a beam of circularly polarised light. *Proc. Roy. Soc. London A 82*, 557 (31 July), 560–567.

PYTHON. Python Programming Language. Web site. http://www.python.org.

SCHROEDER, W., MARTIN, K., AND LORENSEN, B. 2003. *The Visualization Toolkit An Object-Oriented Approach To 3D Graphics*, 4 ed. Kitware, Inc. publishers.

SQUILLACOTE, A., 2008. The ParaView Guide.

THIDÉ, B., THEN, H., SJÖHOLM, J., PALMER, K., BERGMAN, J. E. S., CAROZZI, T. D., ISTOMIN, Y. N., IBRAGIMOV, N. H., AND KHAMITOVA, R. 2007. Utlilization of photon orbital angular momentum in the low-frequency radio domain. *Phys. Rev. Lett. 99*, 8 (22 August), 087701.

VOORS, A. NEC based antenna modeler and optimizer. Web site. http://home.ict.nl/ arivoors/.

Posters

Real-time Global Illumination of Static Scenes with Dynamic Lights (Work in Progress)

Magnus Burénius
magnus.burenius@gmail.com
KTH

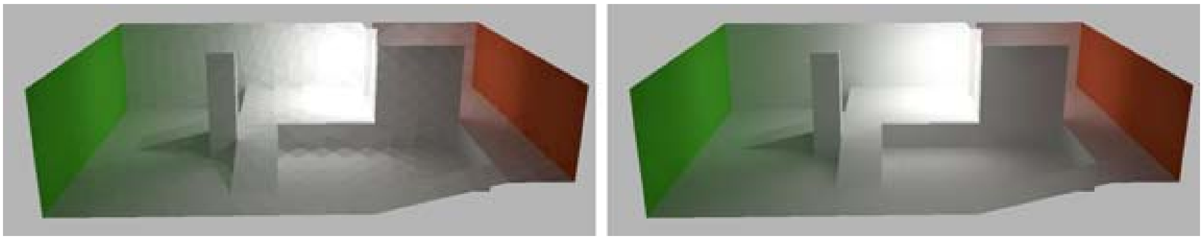


Figure 1: *Indirect illumination of a simple scene. Left: patches for which the light is calculated. Right: linear interpolation between patches.*

Abstract

We describe how the radiosity method can be used to compute the global illumination of a static scene with dynamic lights. A transfer function that maps the direct illumination to the global illumination is precalculated. Real-time performance is reached for simple scenes.

Keywords: global illumination, radiosity, real-time, precomputed radiance transfer

1 Introduction

The radiosity method can be used to compute the global illumination, direct and indirect lighting, of a scene that consists of ideal diffuse surfaces. The surfaces are first divided into patches. The radiosity equation system describes the relation between the global illumination of all patches $B = (B_1, \cdots, B_n)^\mathsf{T}$ and the emission of all patches $E = (E_1, \cdots, E_n)^\mathsf{T}$ [Hanrahan et al. 1991]:

$$MB = E$$

The matrix M depends on the geometric relation between the patches (form factors) and the patches' colors (diffuse reflectance). For our purposes we may consider E to be the direct illumination of the patches. The direct illumination of each patch can be computed in real-time using standard methods like shadow mapping [Akenine-Moller and Haines 2002]. The inverse of M describes the linear mapping from direct illumination to global illumination:

$$B = M^{-1}E$$

It holds coefficients that tell how much light that travels from one patch to another after an infinite number of bounces in the scene. The matrix M is very expensive to compute but if the geometry and textures of the scene are static, M remains static as well. M^{-1} may thus be precomputed and used as a transfer function that maps arbitrary direct illumination to global illumination. The process of computing the global illumination can thus be performed with a large matrix-vector multiplication. That is, for each patch that we want to illuminate globally, we sum over all other patches and accumulate their contribution. Illuminating all patches thus results in a time complexity of $O(n^2)$, where n is the number of patches.

2 Compressing the Matrix

For larger scenes a time complexity of $O(n^2)$ is not manageable. [Hanrahan et al. 1991] reduces the number of interactions, described by M, by clustering of small interactions. This method can be modified and used for M^{-1} as well. It however only clusters neighbouring patches lying in the same plane. For scenes consisting of large planar surfaces the time complexity is reduced to $O(n)$.

[Willmott et al. 1999] describe a method that extends clustering to patches that are just approximately planar. They use it for the radiosity equation system described by M. Extending it to M^{-1} is not as straight forward as the planar clusters first mentioned. This and other techniques are discussed by [Lehtinen et al. 2008]. It should also be noted that it is just necessary to update the interactions between patches visible on screen and those illuminated directly.

3 Conclusion

It is today possible to do real-time global illumination of simple static scenes with dynamic lights. There exist promising ideas and if they are all incorporated in an implementation, preferably accelerated by graphics hardware, real-time performance will probably also be possible for more complex scenes.

References

Akenine-Moller, T., and Haines, E. 2002. *Real-Time Rendering*. A. K. Peters, Ltd., Natick, MA, USA.

Hanrahan, P., Salzman, D., and Aupperle, L. 1991. A rapid hierarchical radiosity algorithm. In *SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 197–206.

Lehtinen, J., Zwicker, M., Turquin, E., Kontkanen, J., Durand, F., Sillion, F., and Aila, T. 2008. A meshless hierarchical representation for light transport. *ACM Trans. Graph. 27*, 3.

Willmott, A., Heckbert, P., and Garland, M. 1999. Face cluster radiosity. In *Rendering Techniques '99*, Springer Wien, New York, NY, 293–304.

Design and Implementation of a Stereoscopic Display in a Lecture-room

Martin Ericsson*
UPPMAX
Uppsala University

Anders Hast†
Creative Media Lab
University of Gävle

Ingela Nyström‡
Centre for Image Analysis
Uppsala University

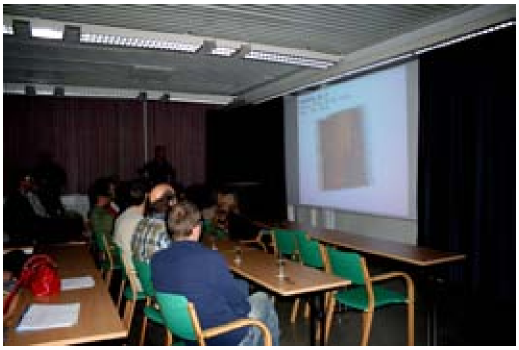


Figure 1: A view of the stereo screen used in the project described in this paper.

Abstract

This paper describes the master thesis project *3DIS4U: Design and implementation of a distributed visualization system with a stereoscopic display* carried out at Uppsala University. The main contributions of the thesis are the installation and evaluation of a wall-sized stereoscopic display in a class room-like environment and improvement of the quality, interactivity and usability of visualizations at Uppsala University by connecting the system to one of UPPMAX high-performance computing (HPC) clusters. The project involved modifications to open source softwares, mainly the Visualization ToolKit (VTK) and ParaView. Furthermore, software was developed to aid users in creating interactive stereoscopic simulations. Software was installed and modified for better usability. The option of using HPC resources for larger interactive visualizations also exists. As a final step, evaluations of the display and of the software were carried out together with background research on distributed rendering techniques to be able to produce a proposal for further development of the project. The result of this work is a class room environment which in a few minutes can be turned into a visualization studio with a stereoscopic display with the ability to create interactive visualizations. The lecture room retains its function as a class room and can support up to 30 simultaneous viewers.

Keywords: interactivity, stereoscopic displays, distributed visualization, high performance computing

1 Introduction

The Centre for Image Analysis (CBA), at Uppsala University and the Swedish University of Agricultural Sciences, has recently acquired equipment for building a stereo projection wall. The project has a close collaboration with Uppsala Multidisciplinary Center for Advanced Computational Science (UPPMAX) that will provide access to high-performance computing (HPC) clusters for performing visualizations on. How do we setup this in a way that it becomes available to as many users as possible? This question is answered in detail in [Ericsson 2008] and briefly presented here.

A prerequisite is that the system should be installed in a class room environment that supports up to 30 simultaneous users (see Figure 1). The room should retain its function as a lecture room while at the same time be able to switch to stereo projection in a matter of minutes. The hardwares used in this project are two workstations, one with AMD dual-dual-core processors equipped with a high-end graphics card and one workstation with a two Intel quad-core processors and a mid-range graphics card. One of the workstations has a 2Gbit/s link to a 200 node cluster provided by UPPMAX.

Software development is also part of the project where suitable solutions are to be found, setup and modified when necessary. The solution should strive to be as usable as possible but at the same time have good performance to allow for interactive visualizations. The expected user of the system does not necessarily have a computer science background as the facility is open to all researchers and students at the university.

*e-mail: maer6521@student.uu.se
†e-mail:aht@hig.se
‡e-mail:ingela.nystrom@cb.uu.se

2 Implementation

The stereoscopic display is using spectral multiplexing, so called Interference Light Technology (INFITEC) [Jorke and Fritz 2003], to mediate improved depth perception and color representation. Spectral multiplexing works by dividing the visual spectra into bands, two bands for each of the primary colors red, green and blue. These bands are then separated by filters and divided so that one band of each type reaches each eye, i.e., half of the red spectra reaches the left eye and the other half reaches the right eye. The filters are mounted onto the two projectors used in the system as well as in a pair of lightweight glasses that the user wears. The INFITEC technology fits well into the project as it provides a solution for multiple viewers both with good image quality and at a reasonable price. Two possible alternatives would have been a temporal multiplexing solution or a polarized light based solution. Temporal multiplexing uses a pair of active glasses which is a bit heavier than glasses used in passive methods, which was undesirable for this project. Reasons for not choosing a polarized based method were the lower image quality and the need for a specialized screen.

One of the software solutions chosen is the open-source Visualization ToolKit [Schroeder et al. 2006] (VTK) by KitWare. VTK was modified for additional support for side-by-side rendering, which interleaves both the left and the right view on the projector screen at the same time. This way of rendering does not conform well with the underlying rendering of the normal window manager; the users see both the left and the right view at the same time and the interaction with the mouse pointer can be confusing as the user must keep track of on which desktop it currently resides. Our modification automatically changes from the normal clone mode setup where both projectors shows the same image, which is easier to navigate in, to side-by-side rendering when entering stereo mode.

ParaView [Squillacote 2006] also by KitWare was chosen as the primary software for visualizing larger data sets for this project. ParaView is developed for larger scale visualizations built upon VTK, uses the Message Passing Interface (MPI) for interprocess communication and comes with a mature user interface. ParaView was compiled and tuned for the cluster that we are using to achieve as high interactivity as possible for larger scale visualizations.

Another aid for converting existing applications for use with the stereo wall was also developed. A small library with help functions meant to be used as guidelines when porting an application source code that the user is well oriented with. The library is written in C and uses OpenGL for rendering.

3 Results

The, by today's standard, relatively low resolution (1024 × 768) of the projectors was evaluated by inspection. Users of the display share the opinion that there is no noticeable artifacts due to the resolution. A similar conclusion is drawn about the brightness of the display. The projectors have a listed brightness of 3500 ANSI lumen and three different test scenes where setup under three different lighting conditions to explore under what conditions the display is deemed usable. The display is as a normal projector dependent on the brightness of the material it is displaying and on the lighting condition in the room. The stereo effect is perceivable under all of the tested light conditions, but deemed most comfortable when the dim lights in the lecture room is turned on. What was noted during the tests was that the fluorescent light in the room creates a very disturbing effect in combination with the filters in the glasses. In fact, it produces a green and pink flicker, although the perception seems to vary from individual to individual. A spectral sample of the light gives some hint to why this might be happening. The nature of which the light is produced in a fluorescent light creates spikes which coincide with our filters. Replacing the light with regular light bulbs or a LED light with a more even spectral distribution solves this problem. The projection setup is optimized for the person sitting in the middle of the room, all other viewers get a mildly skewed perspective. The perspective distortion has not been commented on by any of our test users and we consider this effect to be minor. All in all we are very satisfied with the quality of the stereoscopic display.

The experience with the different software used was good and users have created both interactive and pre-rendered visualizations for displaying on the stereo wall. Users with prior knowledge of VTK could port their visualization for displaying on the stereo wall within minutes. With ParaView our users could visualize raw data volumes interactively that before was needed to be down sampled due to performance problems. The guide lines that were presented as a small code library have been used to port existing OpenGL application for the wall without much effort. A non-invasive method of porting applications to stereo was also investigated, the Chromium library, but was not used in this project.

4 Conclusion

We have designed and implemented a lecture room with a stereoscopic display. We have produced guide lines for using the stereo wall and also produced and modified software to help users port their own software to the stereo wall. By distributing the data processing as well as rendering computations onto larger computational resources, we have made it possible to interactively visualize larger sets of data than previously possible for our users.

Future possible directions for the project have been identified. And further development is important in three areas: remote, collaborative and distributed/parallel visualization. These areas are described in the master thesis report [Ericsson 2008].

The stereo wall is fully functional today (see Figure 1) and in regular use in our visualization studio called *Three-Dimensional Image Studio for Uppsala* (3DIS4U). The abbreviation *3DIS4U* can also be read as *3D is for you* which underlines the goal with the project, an accessible stereo wall for all users at the university.

Acknowledgments

Professor Ewert Bengtsson, Centre for Image Analysis, Uppsala University is acknowledged for scientific support. The staff at UPPMAX is acknowledged for sharing their knowledge on clusters and parallel implementations.

References

ERICSSON, M. 2008. *3DIS4U: Design and implementation of a distributed visualization system with a stereoscopic display*. Master's thesis, Uppsala University.

JORKE, H., AND FRITZ, M. 2003. INFITEC — a new stereoscopic visualisation tool by wavelength multiplex imaging. *Electronic Displays* (September).

SCHROEDER, W., MARTIN, K., AND LORENSEN, B. 2006. *The Visualization Toolkit, An Object Oriented Approach to 3D Graphics*. Kitware Inc.

SQUILLACOTE, A. H. 2006. *The Paraview Guide, A Parallel Visualization Application*. Kitware Inc.

Collaborative Live Multimedia Surface Interaction

Rikard Lindell
Mälardalen University
School of Innovation, Design and Engineering
Box 883
721 23 Västerås Sweden
+46(0)21-15 17 59
rikard.lindell@mdh.se

Abstract

This poster presents the construction of an interactive prototype for collaborative live multimedia performances whose purpose is to explore "surface interaction" in practice. The prototype is currently being field-tested. Collaborating laptop musicians and video artists use the prototype in real performance situations using monophonic touch screen or keyboard-and-mouse. The driving vision is that information content is the base for all interaction between users and the systems. I call this concept content-centric interaction. Users conduct their activities in an unbroken creative flow. The computer is a surface onto which all the users' information content is visualised; the surface can extend to infinity like a magic paper. Surface interaction permits content-centric computing, where content of different data types is moulded into blended media. Longitudinal studies of the prototype requires usable features for collaborative music and video live performances delivered by robust technology. The prototype utilises for example OpenGL for all graphics, operating system services were provided by SDL (Simple DirectMedia Layer), and RakNet was used to develop realtime multiuser collaboration services. The game development community's technologies were applicable in creating a non-generic interaction artefact. Surface interaction is well suited for exploring novel interaction techniques, for instance gesture control and multi-touch displays, and is quickly emerging from a concept to usable tools and systems.

3D Urban Landscape Visualization

Zoja Veide[*]
Department of Computer Aided Engineering Graphics
Riga Technical University

Veronika Strozheva[†]
Department of Computer Aided Engineering Graphics
Riga Technical University

Modris Dobelis[‡]
Department of Computer Aided Engineering Graphics
Riga Technical University

1 Introduction

Present time features rapid advancements in information technologies, which are used to communicate architectural design projects. Most of users are primarily interested in models of buildings, terrain, vegetation, and traffic networks. A European Organization for Experimental Photogrammetric Research survey on city models showed that 95 % of participants were most interested in 3D building data. Computer technology, computer graphics, and computer-aided design (CAD) now offer powerful tools for creating and visualizing digital models of cities. Manual measurement and entry are impractical; so researchers use various sensors to acquire accurate data for 3D urban landscapes. They then integrate the resulting 3D building models into spatial databases and geographic information systems (GIS) to support urban planning and analysis applications. The complexity of application demands and technology challenges make urban modelling an intensive research area.

The investigation is aimed at the evaluation and introduction of affective 3D visualization and communication media using concurrent engineering approach in architectural education process at the Riga Technical University. The innovations in architectural design practice in Latvia will facilitate the increase of design quality and productivity considerably. Our review examines existing modern software tools, which enable to solve the tasks of 3D city modelling. In addition paper reviews the recent history of urban landscape visualization technologies and software techniques, outlines some of the issues for representation of the urban landscape elements, as they may be visualized using GIS software.

2 Digital Urban Landscape Visualization

Digital landscape visualization has a relatively short history in the context of other forms of landscape representation – arguably, the first efforts were in the 1960's. The development of CAD and computer graphics in general also started at that time, but the majority of those early efforts were focused on the representation and visualization of objects, such as gears, airplanes, and etc. A specific concern for the urban landscape was present early on in the development of flight simulation software, and it was during the formative early years of GIS development that visualization of terrain, for example, became a subject of study and development, and grid meshes and TINs, among other useful techniques, were invented.

For the next twenty or so years, terrain representation and visualization was predominantly the purview of 'GIS'-style software, with some minor efforts in civil-engineering or computer aided architectural design software. Today, 40 years later, landscape visualization has entered mainstream efforts in professional fields such as architecture, landscape architecture, civil engineering and Hollywood movie-effects, and is now enabled by many CAD and animation/rendering systems, as well as GIS and remote-sensing software. In all of these and other efforts, recent developments in computer science and computer graphics have made breathtaking and eye-tricking effects possible. CAD, GIS, image processing and even digital video technologies and techniques have blurred together into a powerful combined system for creating digital urban landscape visualizations.

3 3D Urban Landscape Visualisation software

Imagery is far more than pictures of the earth's surface. It is a valuable source of data that captures actual events at specific times and places in the world so that you can study how the earth changes over time. ERDAS IMAGINE by Leica Geosystems gives the tools to manipulate and understand this data. ERDAS IMAGINE is a broad collection of software tools designed specifically to process imagery. It allows extracting data from images like a seasoned professional, regardless of experience or education. ERDAS IMAGINE introduces enterprise-enabled geospatial imaging processing. A relational database provides enormous benefit by enabling end-user visibility into the data it contains and increasing the accessibility of the data. This maximizes the investment in image and feature geospatial information. The three modules providing enterprise capabilities are IMAGINE Essentials, IMAGINE Enterprise Loader and IMAGINE Enterprise Editor.

The first tier of the ERDAS IMAGINE suite, IMAGINE Essentials, offers the basic tools for image mapping, visualization, enhancement and geocorrection. It is a powerful, low-cost image mapping and visualization tool that allows combining different types of geographic data with imagery and organizing it efficiently for projects. At the mid-level tier of the ERDAS IMAGINE suite, IMAGINE Advantage builds upon the capabilities of IMAGINE Essentials to offer more advanced and precise mapping and image processing capabilities. IMAGINE Professional, the highest tier of the ERDAS IMAGINE suite, provides a comprehensive set of tools for advanced geographic imaging, remote sensing and GIS professionals. IMAGINE Professional builds on all of the capabilities of IMAGINE Essentials and IMAGINE Advantage with advanced modelling and analysis features. The add-on modules for ERDAS IMAGINE can be grouped into convenient product tiers, with each tier offering your organization a specific level of flexibility to more

[*] zv@neolain.lv
[†] vs@pit.lv
[‡] dobelis@latnet.lv

accurately meet your needs and requirements. Add-on modules: ATCOR is atmospheric correction and haze removal software used to correct changes in the spectral reflectance of materials on the earth's surface. The IMAGINE Vector module allows you to import and export vector data, and clean and build topologically within an ESRI Arc structured format, without conversion. IMAGINE Radar Interpreter provides the fundamental tools needed to process and enhance SAR images. Because it is data source independent, it allows you to work with any SAR imagery. IMAGINE VirtualGIS is a powerful yet easy-to-use visual analysis tool that offers GIS functions and capabilities in a 3D environment. Beyond simple 3D renderings and basic fly-troughs, it allows creating accurate 3D interpretations of projects for interactive presentations. The IMAGINE Subpixel Classifier is a multispectral imagery exploitation tool which detects materials that occupy less than 100% of a pixel and provides an estimate of the amount of material present. The IMAGINE Developers' Toolkit consists of a set of libraries and documentation that allow you to customize and extend ERDAS IMAGINE. Stereo Analyst is a 3D model generation, interpretation, measurement and visualization tool, which uses stereo imagery to derive 3D information. It is used to create 3D models that are output to IMAGINE VirtualGIS for presentation.

ArcGIS by ESRI is an integrated collection of GIS software products for building a complete GIS. There are four products; each adds a higher level of functionality. ArcReader is a free viewer for maps authored using the other ArcGIS Desktop products. It can view and print all maps and data types. It also has some simple tools to explore and query maps. ArcView provides extensive mapping, data use, and analysis along with simple editing and geoprocessing capabilities. Arc Editor includes advanced editing for shape files and geodatabases in addition to the full functionality of ArcView. ArcInfo is the full function, flagship GIS desktop. It extends the functionality of both ArcView and ArcEditor with advanced geoprocessing. It also includes the legacy applications for ArcInfo Workstation.

Using optional extensions with ArcGIS Desktop products allows you to perform extended tasks such as raster geoprocessing and three-dimensional analysis. Unless noted, extensions can be used with ArcView, ArcEditor and ArcInfo. ArcGIS 3D Analyst allows effectively visualizing and analyzing surface data. Using ArcGIS 3D Analyst, can view a surface from multiple viewpoints, query a surface, determine what is visible from a chosen location on a surface, create a realistic perspective image that drapes raster and vector data over a surface, and record or perform three-dimensional navigation. The ArcGlobe application in ArcGIS 3D Analyst allows managing and visualizing, from a local or global perspective, extremely large sets of three-dimensional geographic data. ArcGlobe provides the capability to seamlessly interact with any geographic information as data layers on 3D globe.

ArcGIS Geostatistical Analyst is an extension to ArcGIS Desktop (ArcInfo, ArcEditor, and ArcView) that provides a variety of tools for spatial data exploration, identification of data anomalies, optimum prediction, evaluation of prediction uncertainty, and surface creation. ArcGIS Network Analyst is a powerful extension that provides network-based spatial analysis including routing, travel directions, closest facility, and service area analysis. ArcGIS Schematics enables users to generate, visualize, and manipulate diagrams from data in a geodatabase. Using ArcGIS Spatial Analyst, can derive new information from your existing data, analyze spatial relationships, and build spatial models integrating core ArcGIS Desktop and ArcGIS Spatial Analyst tools. ArcGIS Survey Analyst is an extension to the ArcGIS family of desktop products that allows you to manage survey data in a geodatabase and display survey measurements and observations on a map. ArcGIS Data Interoperability enables ArcGIS Desktop users to easily use and distribute data in many formats. ArcGIS Publisher delivers the capability to easily share and distribute your maps and GIS data. ArcScan for ArcGIS provides a comprehensive, efficient, and easy-to-use set of tools for raster-to-vector conversion. Maplex for ArcGIS is an automated high-quality cartographic text placement and labelling extension for ArcGIS Desktop.

Autodesk Maya, 3D Studio Max, software are the world's most powerfully integrated 3D modelling, animation, effects and rendering solution. Autodesk Maya combines an industry-leading suite of 3D visual effects with computer graphics and character animation tools, enables to realize creative vision for design projects. 3D Studio Max is a professional 3D animation rendering and modelling software package used mostly by game developers, design visualization specialists. Learn tips to create rich, complex design virtualizations or 3D film effects. Autodesk Envision software is an ideal tool for professionals involved in mapping, planning, surveying, civil engineering, and facilities/infrastructure management projects. Autodesk’s latest technologies-, including Autodesk Envision, allow the user to create, share, and manage mapping and design data throughout the project lifecycle. The application works stand-alone on a Laptop PC, a desktop computer, or over the Internet, and is included in both the Autodesk Civil Series and Autodesk Map Series.

Mapper3D - represents means of three-dimensional visualization of the spatial information. Mapper3D it is built in MapInfo Professional and creates the own menu and the panel of tools.

4 Conclusions

The challenges of urban landscape visualization arise in part from the sheer complexity of landscapes – in size, in curviness, in fractal dimension, and so on. Some of the problems come from the need to integrate several different sources of material, or techniques. But to the extent that these challenges can be overcome by ever faster computers, larger disks, cheaper RAM, better software and more clever algorithms.

The considered software is useful modern tool in the field of planning, construction and representations of city landscapes. It is necessary to note, that the given software are means for tasks decisions elaborated under specific problems of the developers of the software given. Separate decisions are necessary for use of the given products for the decision of other problems and furthermore integration with software of other developers for creation of 3D city models.

In spite of the fact that all programs themselves are commercial products, which should be evaluated for their practical availability, it is necessary to solve a problem of more general level. This problem is the software tools integration into general process of planning, construction and representation of city landscapes.

Integration of various products in general 3D visualization of city planning is an important educational and applied problem. Master and doctoral students will be involved in research activities.

Industry session

Algoryx—Interactive Physics

Anders Backman
Algoryx Simulation

Abstract

Algoryx is a startup company standing on two legs: delivering state-of-the-art physics simulation for the simulator industry and developing environments for interactive physics used in teaching and for plain fun. Algoryx is working close to the research environment at Umeå University, which makes it a perfect environment for developing and implementing novel and groundbreaking methods.

Developing accurate simulators used in training and education places high demand on the fidelity and accuracy of the underlying tools.

A physics engine is commonly used for driving the whole simulator which makes it even more important to get consistent fast and stable results.

This talk will give an overview of AgX, a multiphysics engine used in commercial simulators and research and of Phun, a 2D physics sandbox which is free for download and is used all over the world in education and for plain amusement.

Awareness of 3D—rapid market growth

Trond Solvold
Projectiondesign

LISTEN Heavy Demonstrator to illustrate the potential and feasibility of urban soundscape simulation and auralization

Peter Becker & Peter Lundén
The Interactive Institute

Abstract

Our VISION is that future urban planning as well as architecture in general will be both visual and acoustic. In LISTEN project a demonstrator based on acoustic simulation and auralisation is to be developed, prototyping a planning tool for urban soundscapes. By this tool architectural, noise control and sound design solutions for improving urban soundscapes can be auralized at the planning stage.

Peter Lundén and Peter Becker from Interactive Institute, one of the partners in LISTEN project, will present the background and the simulation/auralisation technology for LISTEN Heavy Demonstrator project approved in the first Visualisation call by KK-stiftelsen, Stiftelsen för strategisk forskning, Vinnova, Vårdalstiftelsen and Invest in Sweden Agency.

Link

Link to the program, including a video for LISTEN project:
http://kks.se/templates/ProgramPage.aspx?id=9792