SIACG 2002

1st Ibero-American Symposium in Computer Graphics

Bounding Information and Media Frontiers

2-5 July 2002
Guimarães-Portugal
University of Minho, School of Engineering

Short Papers

Short Papers

SIACG 2002

1st Ibero-American Symposium in Computer Graphics

2-5 July 2002
Guimarães-Portugal
University of Minho, School of Engineering

Organization

Centro de Computação Gráfica

GPCG
Grupo Português de Computação Gráfica

EGSE
EUROGRAPHICS
Sección Española

EG
EUROPEAN ASSOCIATION FOR COMPUTER GRAPHICS

In cooperation with

ACM SIGGRAPH

SBC
Sociedade Brasileira de Computação

Diamond Sponsors

FCT
Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA CIÊNCIA E DA TECNOLOGIA

POSI
PROGRAMA OPERACIONAL
SOCIEDADE DA INFORMAÇÃO

Platinum Sponsor

ICAM
INSTITUTO DO CINEMA AUDIOVISUAL E MULTIMÉDIA

Gold Sponsor

adi
agência de inovação

Silver Sponsors

Câmara Municipal de Guimarães

CENTRO ALGORITMI

innovators
by Optimus

Supporters

AIMinho

INSAT

UniWeb

Centimfe

Cefamol

APDC

Prospero dos Santos
Joaquim Jorge
Aderito Marcos
(eds)

Printed on June 2002
Grupo Portugues de Computacao Grafica
Revista VIRtual / VIRtual Journal
ISSN: 0873-1837

http://virtual.inesc.pt

Local Organizers Foreword

The 1st Ibero-American Symposium in Computer Graphics (SIACG) aims at becoming the first in a series of similar events organised every year, alternating between Europe and South America. Its mission is to foster transatlantic co-operation through the exchange of scholars, ideas, information and participation in joint projects between Computer Graphics communities in the Iberian Peninsula and Latin America.

The local organization of this year's event has been assumed by the Computer Graphics Center (CCG) of the University of Minho. This represents an important step in the process of CCG's establishment in the Minho Region as a novel R&D Interface institute of the University of Minho while promoting Computer Graphics locally in close cooperation with INI-GraphicsNet - the largest world-wide network of R&D institutions in Computer Graphics and Applications. This framework justifies the decision to complement the scientific and technical programme with an Industry Forum, therefore to stimulate a lively debate about the potentialities of Computer Graphics in different industrial sectors. By bringing together the leading people from the industry, user communities and academia, the forum will allow them to discuss good practices, current and future challenges and opportunities for practical applications of Computer Graphics, which has matured into a transversal application area. The Forum includes a special session assumed by the Portuguese Agency for Innovation in collaboration with the European Commission – DGXIII and the @Lis Program with the aim to widely disseminate existing opportunities in Industrial R&D involving teams in Portugal, Spain and Latin America.

This year's Symposium put together also other important events, in parallel to the technical and scientific programme, such as an Exhibition Fair, Lab Presentations and the Alfa Open Workshop. SIACG is held in the Campus of the University of Minho in the beautiful city of Guimarães – the first Portugal's capital city where the country was born nine centuries ago. Its historic, artistic and monumental treasures have been recently recognised by UNESCO as World Heritage, which constitutes a unique place to visit and enjoy.

We should mention here the dedicated effort of many people that have made possible the local set up of SIACG. First we thank Joaquim Jorge for his personal encouragement and precious help in designing and implementing the various events included in SIACG. Second, we would like to thank all the CCG staff and student volunteers who spent many days (and sometimes also nights) to make sure the organization ran smoothly. Third, we thank the chairs of the SIACG committees specially Manuel Próspero dos Santos, Augusto Sousa and Miguel Dias for their suggestions, ideas and concrete support in providing information, contacts and proofreading texts.

Finally, we would like to send a special thanks to the School of Engineering and the Department of Information Systems of the University of Minho for the rooms, mailing and other facilities made available to the Organization of SIACG 2002 as well as all the sponsoring institutions for their support. Without their effective cooperation and active support SIACG 2002 would not be a reality.

Guimarães, July 2002

Adérito Fernandes Marcos

Alphabetical Index

A Technique for visualization of contextual 4D event related functional magnetic resonance

Pedro Paulo de M. Oliveira Jr
Informática – InCor – HCFMUSP
Dep. Eng. Sistemas Eletrônicos – Escola Politécnica
Universidade de São Paulo
Av Dr Enéas de Carvalho Aguiar, 44
São Paulo - Brasil
ppm@incor.usp.br

Edson Amaro Junior
Institute of Psichiatry
King's College
University of London
Instituto de Radiologia
HCFMUSP
sphaeam@iop.kcl.ac.uk

Michael Brammer
Institute of Psichiatry
King's College
University of London
m.brammer@iop.kcl.ac.uk

Roseli de Deus Lopes
Dep. Eng. Sistemas Eletrônicos – Escola Politécnica
Universidade de São Paulo
roseli@lsi.usp.br

Abstract

This paper describes a new visualization technique for event related functional magnetic resonance imaging (ERfMRI). ERfMRI is a class of functional brain analysis. Although ERfMRI is very powerful in modelling experiments to map brain functions it suffers from one deficiency, the difficulty to extract and analyse all information the exam provides. In this paper we outline this visualization technique, show some results and discuss how this visualization technique can be improved both for research and clinical applications.

Keywords

Event related fMRI, visualization, medical computer graphics.

1. INTRODUCTION

Functional magnetic resonance imaging (fMRI) is a non-invasive technique to map the brain.

fMRI uses the natural contrast provided by the deoxyhemoglobin/oxyhemoglobin ratio. Deoxyhemoglobin is paramagnetic, whereas oxyhemoglobin is diamagnetic. Since their magnetic property is different, areas with higher concentration of deoxyhemoglobin are depicted as areas of lower signal in magnetic resonance imaging (MRI) T2 weighted images. The images produced using the oxyhemoglobin and desoxyhemoglobin contrast is called BOLD (Blood oxygenation level dependent) image.

The BOLD effect is also influenced by an increase in the blood flow to the active brain tissue. The exact underlying mechanism of the BOLD effect is still unknown. For more detail see [Kwong1992], [Ogawa1993].

The increase of MRI signal due to the BOLD effect range from 0.5% to 5%. This is a major challenge posed to the analysis technique because the noise level range is also very similar.

The time serie analysis is frequently based on a calculation of a parametric statistical map that uses the assumption of a Gaussian distribution. An alternative option is to use non-parametric methods where the assumption is that the distribution is unknown.

fMRI studies are based on an experimental paradigm construction that will be evaluated through the hemodynamic response of the brain. There are two ways of creating these paradigms for fMRI, the block design and the event related design.

The block type study consists on an image acquisition serie while the subject executes some task sequentially during a time frame and a control task during the next period. There are variations in the block design with multiple tasks. Another type of design is the event related study, which will be described in section 2.

There are some published papers about fMRI post processing in order to register images as for instance [Pizzi2001] and [Woon99]. Interestingly, there are very few studies regarding the aspects involved in providing better visualization for the the data, especially in erfMRI, where the time component provides an extra dimension to the results.

2. EVENT RELATED fMRI (ERfMRI)

Event related is a recent fMRI design. The experiment with event related fMRI paradigm is based on the observation of the BOLD effect after a short stimulus. The main advantage of ERfMRI is that this technique is less influenced by cognitive bias frequently present in the block design, such, such as fatigue when performing a continuous task (habituation effects) and as the experiment evolves, subjects tend to form strategies to accomplish the task (learning effects), and can feel tired in the last phases of the block exam, decreasing the cooperation level.

The design of an erfMRI study involves the presentation of a brief stimulus to the subject, like for example, a light flash, a picture shown or even listening to a single word. After this stimulus, images are acquired, and then after a certain period of time that varies between 5 and 15 seconds, we can present the stimulus again. The analysis strategy is to correlate the stimulus with the expected BOLD response in MRI.

The event related design is very powerful also in its temporal information. With event related fMRI we are able to detect areas in the brain that had a hemodynamic response which peaks prior other areas. Figure 1 and Figure 2 show two BOLD curves of different brain areas activated by an auditory presentation or a 300 Hz sinusoidal tone. One can notice that the frontal area of the brain has a hemodynamic peak response around 3 seconds and in the left temporal region the response peaks around the 5th second.

3. VISUALIZATION IN ERfMRI

Although ERfMRI can show this important information about activation times during tasks, there is no efficient way to present this kind of information. Sometimes it is rather difficult to understand the brain circuitry and its relation to an ERfMRI experiment using conventional 3D or even 4D visualization techniques. Figure 3 shows an example of a simplified ERfMRI showing the difficulty to represent the activation in 2D images.

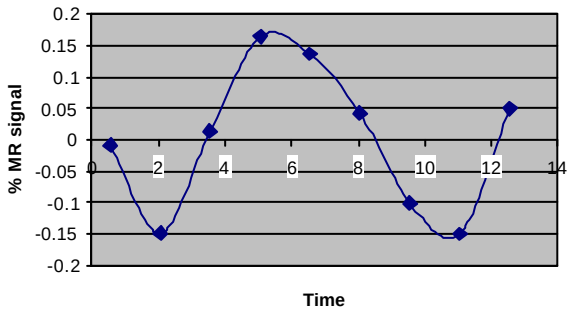


Figure 1 - BOLD curve of the left temporal area

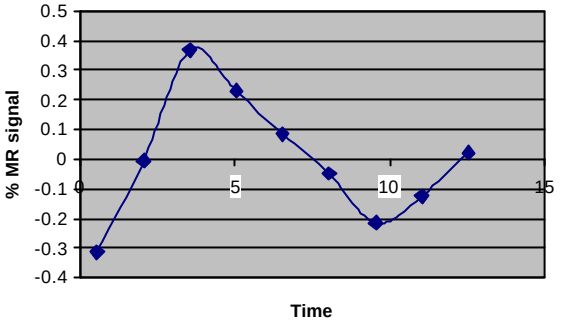


Figure 2 - BOLD curve of the lower right frontal area

4. A NEW 4D VISUALIZATION TECHNIQUE

The technique proposed in this paper is based on a contextual approach for the 4D visualization of the ERfMRI[Oliveira2001].

The 4D ERfMRI visualization algorithm is shown below:

1. Fusion of activation map with high resolution MRI
2. For each time t do
   a. Find activated clusters B for time t
   b. Calculate camera position t to show B
   c. Volume render scene

There are some approaches to solve step 1. We can convert the activation map into Tailarach [Tailarach88] coordinates and make the registration with a brain atlas.

For the step 2a we need to detect the hemodynamic response peak time into each BOLD curve and the respective activated clusters.

Step 2b and 2c are straightforward. A VolumePro VP500 board was used [Pfistner99] in order to achieve real time interactivity in volume rendering large datasets. With this board and a 256Mbytes dataset we can render 30 frames per second.

5. RESULTS AND DISCUSSION

Figure 4 shows an example of ERfMRI study with this new technique of visualization. In spite of the fact that in a printed paper we cannot reproduce the dynamic component, which is the key idea of this work (animated images) Figures 4, 5 and 6 provides an idea of how this type of visualization approach can improve the temporal comprehension of ERfMRI.

A functionality that should be incorporated in a future implementation is to show the cluster B in a time t only if t is the time when cluster B reached its peak hemodynamic

response, or even color-code the clusters providing an intuitive index of activation (i.e. when the cluster is showed in yellow, that time corresponds to the peak).

We should also develop a metric to evaluate which visualization technique is the best suitable for ERfMRI.

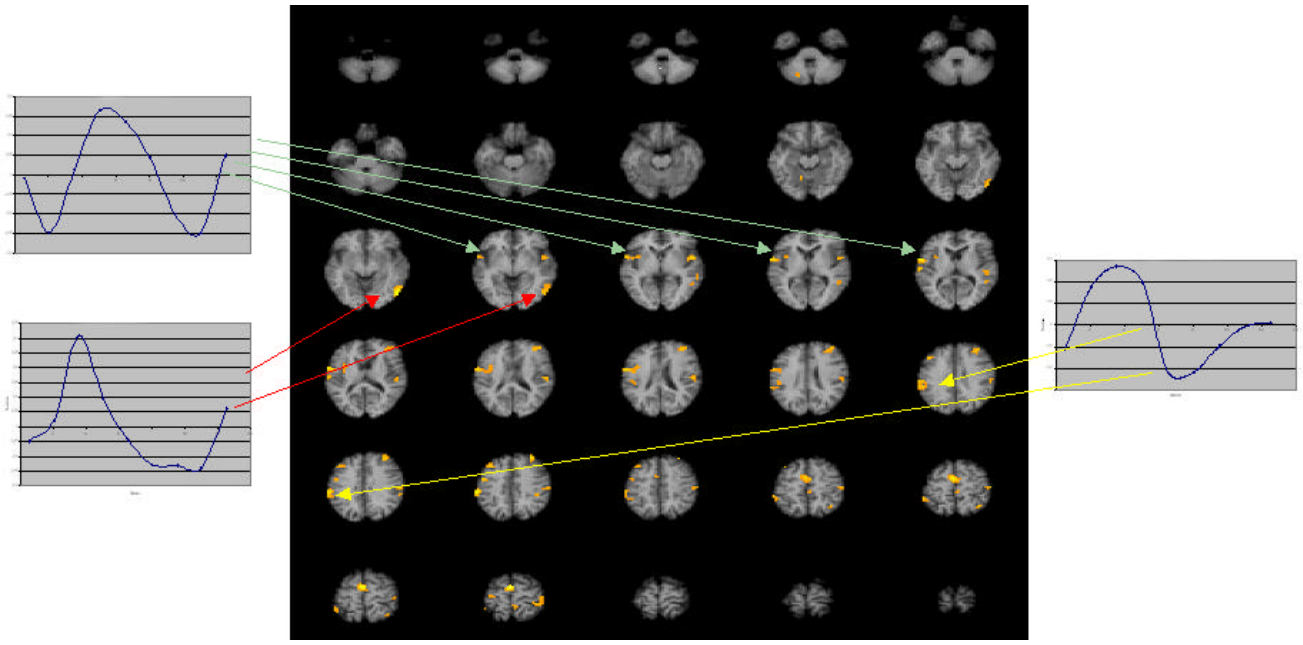


Figure 3 - The display of an ERfMRI experiment

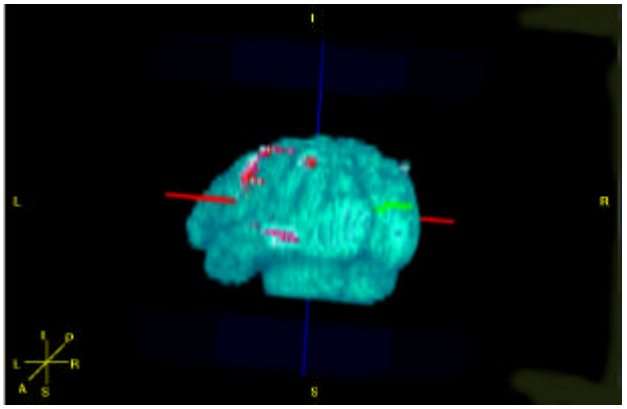


Figure 4 - Activation in t=3s

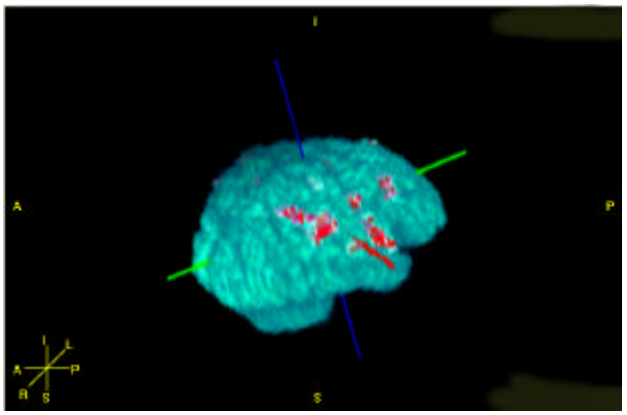


Figure 5 - Activation in t = 4s

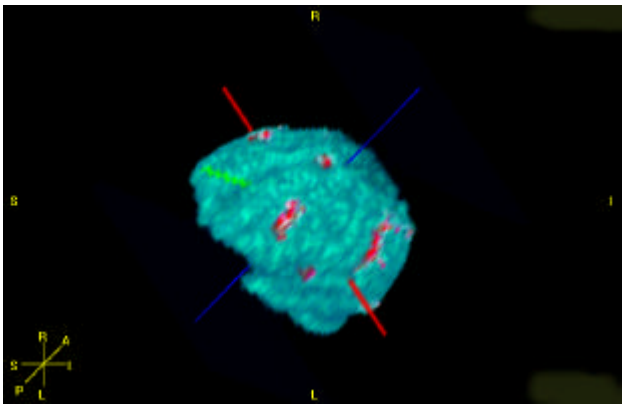


Figure 6 - Activation in t = 6s

6. ACKNOWLEDGEMENTS

We thank all Dr. Umberto Tachinardi, Sergio Furuie, Marcelo Zuffo and Marco Antonio Gutierrez for their contributions and ideias for the development of this paper. This work began at TeCGraf – PUC-Rio and we would like to thank Marcelo Gattass.

7. REFERENCES

[Kwong1992] - Kwong, K. K., Beliveau, J. W., Chesler, D. A., Goldberg, I. E., Weisskoff, R. M., Poncelet, B. P., Kennedy, D. N., Hoppel, B. E., Cohen, M. S., Turner, R., et al. 1992. Dynamic magnetic resonance imaging of human brain activity during primary sensory stimulation. Proc. Natl. Acad. Sci. 89:5675–5679.

[Ogawa1993] - Ogawa, S., Menon, R. S., Tank, D. W., Kim, S. G., Merkle, H.,Ellermann, J. M., and Ugurbil, K.

1993. Functional brain mapping by blood oxygenation level-dependent contrast magnetic resonance imaging: A comparison of signal characteristics with a biophysical model. Biophys. J. 64:803–812.

[Oliveira2001] – Oliveira Jr., P. P. M , Lopes, R. D. , Arantes, P. R. – "A system for event related fMRI post processing" – Proceedings of SIBGRAPI 2001 – IEEE Press. Outubro, 2001.

[Pfistner99] H. Pfistner, J. Hardenbergh, J. Knittel, H. Lauer, and L. Seiler. The VolumePro real time ray-casting system. Proceedings of ACM SIGGRAPH, pages 251–260, 1999.

[Pizzi2001] - Pizzi, N. J , Vivanco, R. A. , Somorjai, R. L. - "EvIdent: a functional magnetic resonance image analysis system" - Artificial Inteligence in medicine 21 (2001) 263-269

[Tailarach88] Talairach, J., and P. Tournoux. 1988. A co-planar sterotactic atlas of the human brain. Thieme, Stuttgart, Germany.

[Woon 1999] - Woon-Young So, Dong-Seok Jeong, "A study on the postprocessing of functional MRI using overcomplete wavelet transform." IEEE TENCON, 1999, pp 1174-1177

Augmented Reality for Non-Invasive Medical Imaging

Paulo Lopes
Paulo.Lopes@iscte.pt

António Calado Lopes
Antonio.Calado@iscte.pt

José Miguel Salles Dias
Miguel.Dias@iscte.pt

ADETTI/ISCTE, Associação para o Desenvolvimento das Telecomunicações e Técnicas de Informática, Edifício ISCTE, 1600-082 Lisboa, Portugal, www.adetti.iscte.pt

Abstract

In this paper we present an Augmented Reality technique that enables the real time mapping of general medical imaging data over a real human body. The system is based on Bezier patches registered over the human's body, allowing the creation of smooth curved surfaces, where the controls points are identified by means of fiducial markers placed over the same body, on user defined zones. Over this surface we then map, in real time, the clinical imaging data corresponding to non-invasive medical imaging modalities, such as echocardiography.

Keywords

Augmented Reality, Bezier patches, AR Toolkit, Medical Imaging.

1. INTRODUCTION

One of the obvious applications of the emerging area of Augmented Reality is in the context of medical imaging. With such an application a hand, the physician, working for example in a non-invasive auxiliary diagnosis setting, such as in ultra sound imaging, could have a view inside the patient by just looking at him. To tackle this scenario, we have developed a computational system that enables the real time mapping of general medical visualization procedures (e.g. as in echocardiography) over a real human body. Still some problems arise: how to map the real time medical data set over the area being examined? How to make an adaptation of a virtual surface to the patient's body? How accurate this approximation is? In order to answer to these problems, we propose a system based on Bezier patches, which allows the creation of smooth curved surfaces, where the controls points are identified by fiducial markers placed over the patient's body, on user defined areas.

The paper is organised as follows: in section 2, we provide a background and state-of-the-art on Augmented Reality and its applications of in medical scenarios. In section 3, we present our technique to achieve smooth registration of a Bezier surface onto a real human body and in real time. Section 4, covers the issues of texture mapping of imaging data on the Bezier surface. Section 5 details the hardware and software platforms and discusses the achieved results of system testing. Finally, in section 6, conclusions are drawn and future directions of research are given.

2. BACKGROUND: AUGMENTED REALITY AND ITS APPLICATIONS

Augmented Reality (or AR) systems and technologies were first introduced in 1992 by Caudel and Mizell [Caudell92], in the context of a pilot project, where they were used to simplify an industrial manufacturing process in a Boeing airplane factory. In general these systems provide the means for "intuitive information presentation, which enhances the perceiver's situational awareness and cognitive perception of the real world" [Behringer98]. It can also be seen as a "knowledge enhancement of the real world" [Seitbert99]. Technically, this enhancement is achieved by computing the virtual camera parameters, that mach the position and orientation of the observer of the real scene. After this step, "virtual" objects or information cues, can then be registered in relation to "real" objects, which means that the first ones, can be seen in the same position and orientation of other "real" objects of the scene, as perceived by the user. From the computational architecture point of view, this is usually done using optical or video see-through head mounted displays and tracking devices, linked to either standalone computers with 3D graphics capabilities, or mobile wearable computers. Video see-through AR is where virtual images are overlaid on live video of the real world. A possible alternative is optical see-through AR, where computer graphics are overlaid directly on a view of the real world. Optical see-through AR has more complex camera calibration and registration specifications, requiring electromagnetic, ultrasonic or hybrid systems to track the position and orientation of the viewer, and a separate system to register the virtual objects in respect to the real ones. Video see-through AR, generally uses optical camera tracking based in either geometrical recognition of well known fiducial markers placed onto the scene, and on the knowledge of the camera internal and external parameters [Kato99], or of geometrical features algorithmically extracted from the image scene, which then enable the computation of the virtual camera parameters [Pollefeys2001]. The Video see-through AR approach has the advantage of using a single data stream to account for the

viewer and scene objects registration. In synthetic terms, Azuma [Azuma97] argues that AR "1) combines real and virtual environments; 2) is interactive in real-time; 3) is registered in 3D. AR is, in fact, a challenging multidisciplinary field and a particularly promising new user interface paradigm, integrating computer graphics, computer vision, positional and orientation tracking technologies, wearable computer architectures, psychology, artificial intelligence, ergonomics and others.

Since 1996, AR and its applications, have been researched by a group of Fhg-IGD, in areas like repair and maintenance of complex facilities, interior and exterior architecture, as well as in medical scenarios, such as needle biopsy and minimal invasive treatment, combining Computed Tomography (CT), Magnetic Resonance (MR) and Ultrasound (U) imaging, with the real scene, enabling news means of interaction of the physician with the patient [Reiners98] [Seitbert99]. Our work focuses currently in the registration of ubiquitous, non-invasive, medical imaging modalities, such as echocardiography, onto the surface of the patient, so as to enable to physician to have a view inside the patent, by just looking at him.

3. ADAPTING A VIRTUAL SURFACE TO A REAL HUMAN BODY

One of the major problems that arise in the mapping of medical imaging data set over the surface of a patient's body is precisely where to map it. Therefore, a virtual surface with approximate physical characteristics to the real body has to be generated, where the real time imaging data is mapped on the latter.

In the following section, an analysis is made on how to generate this virtual surface and the best way to adapt it to a real body.

3.1 The virtual surface

The selected geometrical surface model was the Bezier patch. This type of surface model, allows us to create a smooth surface given a certain number of control points depending on the degree of the polynomial used.

For our work, we have used a Bezier patch based on the quadratic Bernstein polynomial as shown in equation 1:

$$(t + (1-t))^2 = 1^2 \qquad (1)$$

Developing this formula and taking into account that entire left side is equal to 1, it's safe to assume that if we add all the components they should still equal one. Based on these assumptions, we can write the following expression:

$$P_1 t^2 + P_2 2t(1-t) + P_3 (1-t)^2 = P_{new} \qquad (2)$$

where P1, P2 and P3 are three arbitrary control points. If we vary t between 0 and 1, we can generate a morphed curve between P1, P2 and P3. It should be noted that the generated curve is interpolated at the end-points (P1 and P3) and is attracted to P2, never crossing it.

We could question why a quadratic polynomial is used instead of a cubic or higher degree polynomial. The reason is related to the minimal number of control points that we have assumed to be necessary, to accurately describe a physical body surface. With a quadratic Bernstein polynomial, we can generate curves with only three control points, and if we want to create a patch, a 3x3 matrix of control points is needed, which is enough to create a quite smooth and accurate surface. The patch is formed by crossing generated curves in two different directions. In the following figure a 3x3 patch can be seen.

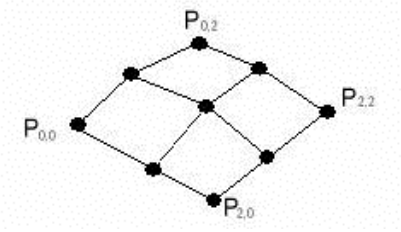


Fig. 1 A 3x3 Bezier patch

3.1.1 Fiducial Markers Tracking and Control points Layout

Having described the way to build smooth accurate surfaces we can now examine how to use them with Augmented Reality.

Our system lies in ArToolkit [Kato99] [Kato2001], a C and Open GL-based publicly available library that uses accurate vision based tracking methods to determine the virtual cameras' viewpoint information through the tracking fiducial markers. First the live video image (Figure 3) is turned into a binary image based on a lighting threshold value. This image is then searched for square regions. ARToolKit finds all the squares in the binary image, many of which are not the tracking markers. For each square, the pattern inside the square is captured and matched against some pre-trained pattern templates (Figure 2). If there is a match, then ARToolKit has found one of the AR tracking fiducial markers. ARToolKit then uses the known square size and pattern orientation to calculate the position of the real video camera relative to the physical marker. In fact, upon detection of a specific marker, ArToolkit provides the programmer with a transformation matrix, that translates and re-orients the local coordinate system associated to the marker, to the virtual camera coordinate system. Since the virtual and real camera coordinates are the same, the programmer can precisely overlay the image of a virtual object onto the real world, using OpenGL, resulting in a Augmented Reality effect. This virtual object can be exactly overlaid on top of the marker or in any position relative to a local coordinate system attached to it.

Basically, we have in our system five fiducial markers, whereas their centre's coordinate positions, corresponds to five control points of the 3x3 Bezier patch (points 1, 2, 3, 4, 5 of Figure 2). All the other remaining four control points, are interpolated from their neighbours (points A, B, C, D of Figure 2). The interpolation is basically a simple average of the positions of the neighbours.

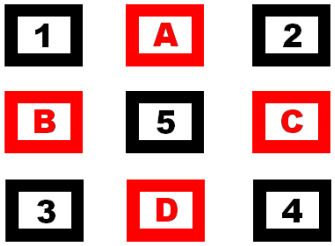


Fig. 2 The maker layout

We can question the adoption of a five marker layout, instead of a nine marker layout. There are two reasons for this: obviously, the nine markers option, does require more computational resources than the five marker's one, without a significant gain in the obtained accuracy of the surface. The other reason is a practical one: while we were testing the different possible layouts, the five marker solution has proved empirically, to be more easily manipulated. The adaptation of the virtual surface to the human body, is very simple: we just physically glue the markers to the physical surface zone, from which we obtain the clinical imaging data that we want to examine. The markers should always respect the layout ordering defined in Figure 2. In Figure 3, we can observe the Bezier patch applied to a physical body.

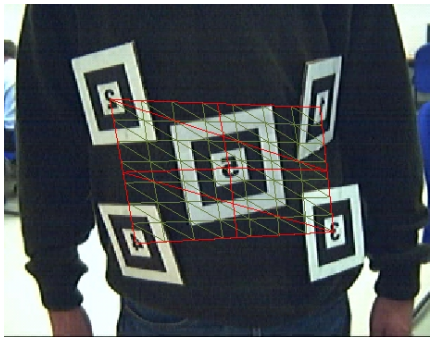


Fig. 3 The virtual surface applied to a physical body

As depicted in Figure 4, the interactive and real-time manipulation of the virtual surface is very simple, and only requires, as expected, the manipulation of the Bezier patch control points.

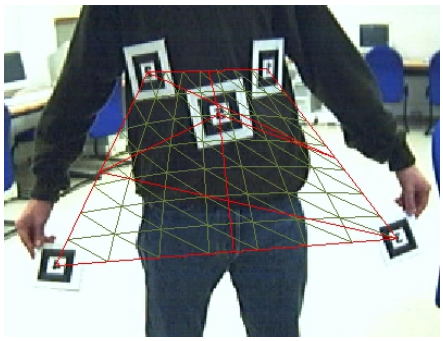


Fig. 4 The manipulation of the virtual surface

4. MAPPING OF THE CLINICAL IMAGING DATA OVER THE VIRTUAL SURFACE

Having defined the virtual surface registered onto the patient body, we can now map the non-invasive clinic imaging data over the latter.

The clinical data is basically the composite video stream of an auxiliary non-invasive diagnosis examination procedure, like an echocardiography. In our system, we simply digitise the composite video input source, capture the digital video frames and map them over the virtual surface by continuously changing the texture, all this in real time (we've achieved 18 fps). Since the Bezier patch is also registered in real time relatively to the patient's body, any motion of this last one, does not affect the augmented examination procedure.

In the next figure, we depict a snapshot of this procedure

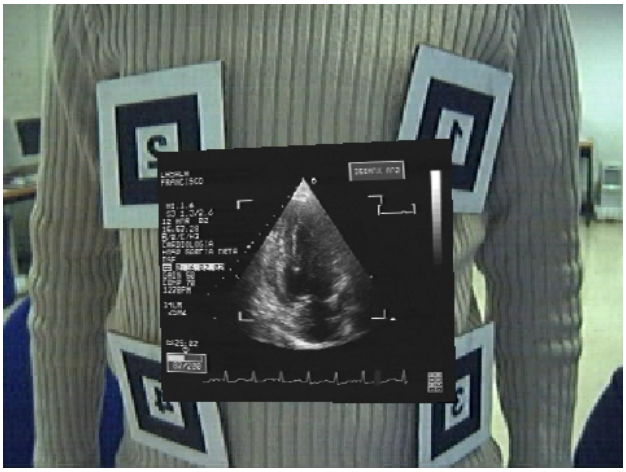


Fig. 5 Mapping of the clinical imaging data over the virtual surface

5. RESULTS AND SYSTEM TESTING

The hardware and software platform used in our system, are as follows:

Hardware:

- CPU: HP Pentium III 1GHZ
- RAM: 256 Mbyte
- Graphics card: NVIDIA GFORCE2 MX 32 Mbyte
- Video camera - 2 options were used: MicroCam wireless ultra miniature camera from Swann Security; and WebCam 5 from Creative Labs
- Video see-through head mounted display: Olympus Eyetrack SMD 700, Multimedia Glasses, which support 800x600 pixel resolution
- Video capture: Optibase real-time composite and S-Video video capture and MPEG1 encoder.

Software:

- MS Visual C++ 6.0 enterprise edition
- ArToolkit 2.5.2
- Video Input: Direct X 8.1
- Graphical output: Open GL

Indoor Tracking method used:

- Computer vision-based approach as provided by ArToolkit with sparsely placed fiducial markers.

After an exhaustive period of testing, the system behaved as we have proposed. By running the application on a Pentium III 1.0 Ghz with a GFORCE2 MX graphics board, we have concluded that the performance of the system was quite smooth, reaching an average of 18 fps. It should be noted that, for each frame, the Bezier patch is totally recalculated, as well as the texture mapping.

Relatively to the accuracy of the surface, we have concluded that the Bezier patch was a good choice; although we had some inconveniences with the centre marker (refer to Figure 2). As mentioned, the Bezier patch is only interpolated at the end points, while marker 5 is only an attractor of the surface, thus some difficulties may arise in generating steep curved surfaces. This can be overcomed, however, by selecting an interpolated surface patch based, for example, in Kochaneck-Bartels splines.

6. CONCLUSIONS AND FUTURE ACTIVITIES

In this paper we have presented a general technique to enhance the visualisation of medical imaging, by overlapping the non-invasive clinical data over the body of the patient in real time. This system opens a wide range of opportunities for the clinical practice, namely in enhancing *multimodal* image registration. In this specific setting, different data sets coming from various medical imaging modalities, ranging from anatomical (MR, CT, Xrays, Ultra-sound) to functional (PET, Positron Emission Tomography, etc.), are realigned in a common reference frame, so that they come in special correspondence, showing both anatomy and function on the same image. Augmenting the *multimodal* image registration with our technique, and mapping correctly the integrated data set in relation to the patient's body, will certainly provide a rich and efficient experience to the medical diagnosis practice, and eventually to the chirurgical as well. A specific project in augmenting echocardiographic images of the left ventricle in association with the Cardiology Department of Hospital Garcia de Orta and Instituto de Fisiologia of Hospital de Santa Maria is currently under way, aiming at performing registration of a 3D geometrical and mechanical model of that organ, with non-invasive echocardiography imaging data. This research aims at evaluating stress-strain of the left ventricle in disease states, such as ischemic cardiomiopathy and heart failure – two of the most prevalent diseases in the developed world. The quantification of registered and augmented physically-based object model of the left ventricle will allow the collection of objective data for research purposes, namely the effects of new therapies, as well as for clinical purposes by increasing diagnostic accuracy. To detect ischemia and to make therapeutic decisions, concerning bypass surgery, the effects of drugs or exercise on the echocardiogram are currently performed. Its results are currently analysed in a subjective way. The new approach will make the current decision process a quantitative-based one, improving diagnostic and allowing objective comparisons to be made.

7. ACKNOWLEDGEMENTS

The authors would like to thank Prof. Manuel Carrageta, head of Cardiology Dept, Dr. Luis Brás Rosário, Cardiologist, both in Hospital Garcia de Orta, Almada, and Prof. Silva Carvalho head of Instituto de Fisiologia of Hospital Santa Maria, Lisbon for their valuable comments and suggestions and the provision of echocardiography videos.

8. REFERENCES

[Azuma97] Azuma, Ronald T. A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments 6*, 4 (August 1997), 355 - 385

[Caudell92] Caudell, T., Mizell, D., "Augmented Reality: An Application of Heads-up Display Technology to Manual Manufacturing Processes", Proc. Hawaaii International Conference on Systems Sciences, Maui, Hawaii, IEEE press, pp 659-669, January 1992.

[Kato99] Kato, H., et al., "ARToolKit, PC version 2.11", Hiroshima City University, December 1999

[Kato2001] Kato, H., "Developing Applications with ARToolkit", SIGGRAPH 2001 Course Notes 27, 2001.

[Pollefeys2001] Pollefeys, M., "Obtaining 3D Models With a Hand-Held Camera", Center for Processing of Speech and Images, University of Leuven, Belgium, Lecture Notes SIGGRAPH Course 2, August 2001

[Reiners98] Reiners D., et al. "Augmented Reality: Applications", *Computer Graphik Topics, reports on Computer Graphics*, 4/98 Vol 10, pp 36-38.

[Seitbert99] Seitbert, F., Hildebrand, A., "Stereo based Augmented Reality applied within a Medical Application", *Computer Graphik Topics, reports on Computer Graphics*, 1/99 Vol 10, pp 24-26.

Automatic Lightmap Generation for Generic Sets of Geometry in Real Time Simulation

Ayucar, Iñaki
CEIT
San Sebastian, Spain
iayucar@ceit.es

García-Alonso, Alejandro
University of the Basque Country
San Sebastian, Spain
agalonso@si.ehu.es

Matey, Luis
University of Navarra
San Sebastian, Spain
lmatey@tecnun.es

Abstract

This paper describes an automatic lightmap method for generic sets of geometry and its use in real time rendered scenes using Hardware Transform and Lighting.

Keywords

Lightmaps, real time visualization.

1. INTRODUCTION

Arvo and Kirk introduced lightmapping [Arvo87] and it has been a popular technique for years. It adds to the scenes pixel-level detail in lighting calculations and gives a way to put into the final result a complex lighting model with no computation overhead at runtime.

Figure 1 Lightmapping Benefits

Lightmaps were first approached in real-time graphics by Carmack, while developing Quake´s lighting model. A small introductory text can be found in [Abrash97].

Last years several improvements have been made, adding support for colour lights and increasing lightmap resolution. As lightmapping is a static lighting technique, other studies targeted on adding dynamic properties to it, like the "Attenuation Maps" method exposed in [Dietrich00]. More ideas can be found in other sources [Thalman91] [Eberly99] [Möller99].

Lightmaps are especially useful when rendering low polygon environments, in which polygon size and, therefore, distance between vertices is quite big. The lack of detail in this kind of scenes when using vertex-based lighting made necessary to look for other solutions like lightmapping (see Figure 1).

There are several ways of building lightmaps. For instance, games often make use of lightmaps for restricted sets of geometry. Lightmaps can also be made using a 3d rendering package. On the other hand, our objective is to develop an automatic ligthmap generator, that works for any kind of object and whose results can be used for real time visualization. This approach saves the tedious work required by the manual method. However, automatic lightmap generation usually requires the creation of an individual texture for each polygon. That structure slows down the performance of real time visualization systems based on the simultaneous use of lightmapping and Hardware Transform & Lighting (T&L).

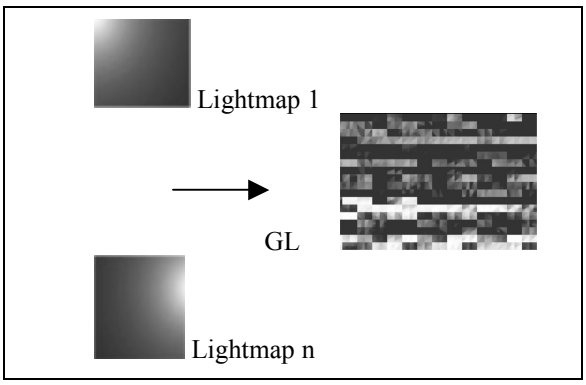


Figure 2 General Lightmap composition

This paper analyzes the data structure and image generation problems that appear when arranging texture maps of several polygons into a unique texture. This structure will be called here the General Lightmap (GL). They are managed in two steps: **Pre-process**, create lightmaps, put them into GL, and tell the triangles which part to take; and **Runtime**, address the GL by each triangle and solve the rendering problems.

Texture coordinates must be assigned to every vertex of

each face, in a way that each face maps its own part of the GL. When loading a model, to automatically set texture coordinates the following information is needed: the GL size in pixels and the number of Rows and Columns that will encompass. The coordinates assigned to the vertices that compose the first triangle should be:

```
LightmapSize_X = GLSize_X / NumOfColumns
LightmapSize_Y = GLSize_Y / NumOfRows
    1.- ( 0, 0 )
    2.- ( 0, LightmapSize_Y / GLSize_Y )
    3.- ( LightmapSize_X / GLSize_X,
          LightmapSize_Y / GLSize_Y )
```

These are for a single-face, but a system to travel through the whole GL should be developed. The calculations will depend on two values: starting pixel of the column currently processed and starting pixel of the row currently processed. A pseudo-algorithm for the texture coordinates generation would be:

```
StartingPixelX = StartingPixelY = 0
For every face in the object
    Assign u,v to vertices
    StartingPixelX += LightmapSize_X
    If EndOfColumn
        StartingPixelX = 0
        StartingPixelY += LightmapSize_Y
```

Coordinates are assigned incrementally. For each face that is processed, StartingPixelX is incremented by LightmapSize_X. If the result is greater than the X size of the GL (end of the column), then it is set to zero and StartingPixelY is incremented by LightmapSize_Y. So, texture coordinates for a given triangle will be:

```
U1 =  StartingPixelX / GLSize_X
V1 =  StartingPizelY / GLSize_Y
U2 =  StartingPixelX / GLSize_X
V2 =  ( StartingPixelY + LightmapSize_Y ) /
      GLSize_Y
U3 =  ( StartingPixelX + LightmapSize_X ) /
       GLSize_X
V3 =  ( StartingPixelY + LightmapSize_Y ) /
       GLSize_Y
```

Here remains unknown which vertices in the triangle are v1, v2 and v3. It will be seen later (Section 3).

2. THE SHARED VERTICES PROBLEM

Geometry is stored using an indexed structure. A face is defined by a list of vertex indices. The definition of each vertex is stored in an array of vertices. With this methodology, vertices that are used by more than one face have a unique instance in memory. So, the amount of data flow and the memory needed to store it are optimized.

Lightmaps are compiled into GL in order of appearance instead of neighbourhood. So, any vertex shared by two faces requires different sets of texture coordinates to access the GL. Although some graphic APIs allow having several sets of texture coordinates for each vertex, the number of faces that share a specific vertex is too variable. So, setting different texture coordinates for a vertex is not recommendable.

The best solution is to parse the geometry at loading time, cloning every shared vertex and modifying the Index List to be non-shared. Although some efficiency in terms of memory management is lost, this way, texture coordinates can be assigned independently for every face. On the other hand, vertex illumination is not penalized, because it is not used. The reason is that, lightmaps replace the need for these computations.

3. U, V EXTENSIONS

It was said above that calculating which vertex extends the U coordinate and which one do it with the V was the way to know how to assign texture coordinates. The meaning of this concept is shown graphically on Figure number 3. When generating the local lightmap for a face, depending on the order vertices are processed, the interpolation of coordinates will follow the positive direction or just the opposite one. This can result on a lightmap flipped horizontally or vertically.

It is obvious that lightmaps must be generated in the same direction that are loaded and the only way to guarantee this is to establish a fixed direction, no matter which one, but always the same. The easiest way to do this is to set a correspondence between the texture coordinates for the primary texture and the coordinates for the lightmap.

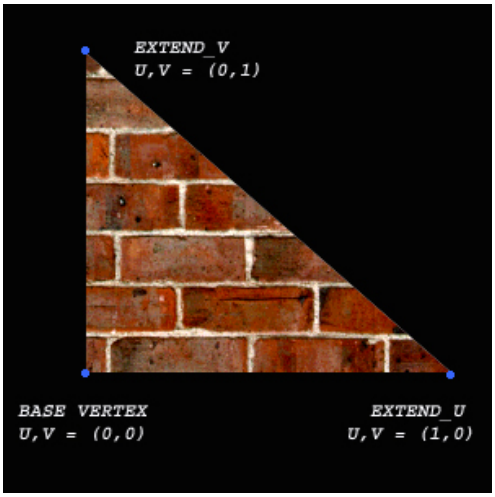


Figure 3 U, V extensions

Both coordinates are 2d, and are easily comparable. If coordinates for the primary texture flow in a positive direction, so lightmaps must do.

4. PRECISION CONSTRAINTS

Now we will explain the need for an algorithm to decide how many sub-divisions must be made in the x and y axis of the GL.

It seems that decision can be made using a simple algorithm like the following one: take the next integer value to the square root of the number of faces of the object. This can be a good idea, but in practice some problems of resolution appear when passing texture coordinates to the graphics API.

If the number of columns is arbitrary a vertex could get a "u" coordinate like 0.125341 or so. 3D-cards cannot achieve that kind of resolution. Some truncation is done and the consequence is the apparition of notorious visual artifacts in the borders of the faces. This effect is aggravated when displaying interactive visualization images. In fact, the precision that use the cards is heavily restricted.

The only way to specify texture coordinates that will remain untouched at rendering is to divide the range 0…..1 in **n** parts, being **n** a power of two. Otherwise, if the GL is divided, for example, in 5 parts a 0.2 coordinate will be generated, value which will not be properly managed by the driver or API at the render stage. Taking this into account, a division algorithm should look like:

```
NbrOfColumns =Next int to sqr( nbr faces )
Counter is initially 0
Repeat a number of times
        Var = 2 ^ Counter
        If Var >= NbrOfColumns
            NbrOfColumns=Var
            Exit
        Increment Counter
```

A more compact expression is :

NC = 2^(ceiling(log_base_2(next int to sqrt(nbr faces)))

5. THE TEXTURE FILTERING ISSUE

Another problem is texture filtering. The final colour that is displayed on a pixel is taken from several calculations like texture rasterizing, lighting and texture filtering.

Figure 4 A Pixelation example

Filtering is done to increment visual quality, decreasing the pixelization-effect shown when we get close to a polygon (see Figure 4). It involves filtering the texel of the texture that corresponds to the pixel actually being drawn with its neighbours.

On the GL, when setting texture coordinates in the borders of a lightmap, the filtering will be done with texels that correspond to another face: there is not image continuity between neighbour lightmaps. The light intensity of the neighbour face may be drastically different, and it will wrongly affect to the actual texel intensity (see Figure 5) producing artifacts on the edge.

This effect is inversely proportional to lightmap resolution, so, two different ways to fix it can be used. Either, increase lightmap resolution until getting acceptable results, or deactivate texture filtering when rendering the lightmap. The last must also include the deactivation of MipMapping. Anyway, as filtering can remain activated for the primary texture, visual quality does not fall much.

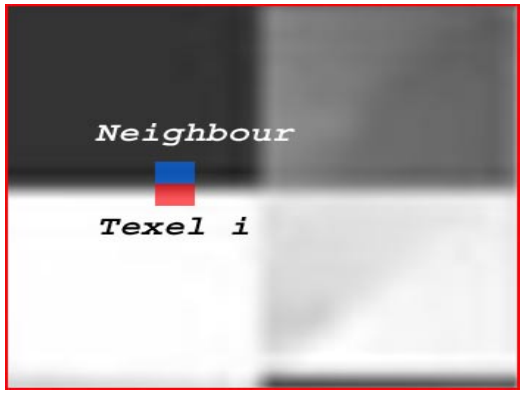


Figure 5 Wrongly averaged texels

6. SAVING MEMORY

Although faces are triangles, up to now the lightmap we have referred to is a rectangle. So, the structure and methods explained follow that assumption. This involves an unnecessary consume of memory which leaves a half of the memory of every GL unused (see Figure 6). A good optimization could be done if adjacency between pairs of consecutive faces could be assured.

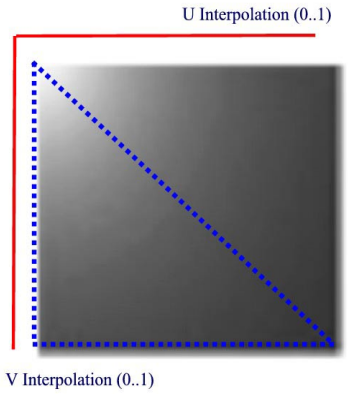


Figure 6 Memory losses in lightmap generation

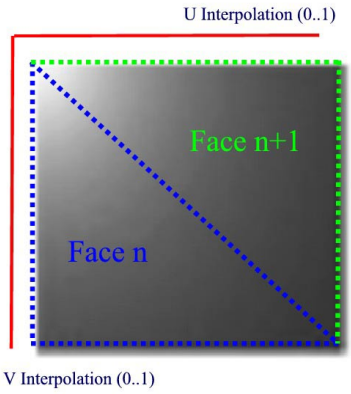


Figure 7 Saving memory

If faces could be arranged in that way, each pair of consecutive triangles will have lightmaps that fit exactly along the shared diagonal. Therefore, memory consume will be reduced and no artifacts will appear along the lightmap shared edge (see Figure 7).

This kind of geometrical restriction is difficult to achieve. However, for some sets of geometry, a pre-process step could be done to fit this purpose. However, some GL cells should store only one lightmap when adjacency cannot be found. To allow this implementation a skip mechanism should be implemented into the list of faces.

7. PRIMARY AND LIGHTMAP TEXTURES

Once the lightmaps have been loaded into the rendering application, the texture coordinates have been generated, and all the previous problems have been fixed out, the rendering process must be considered.

The main factor that has made possible the use of lightmaps in interactive visualization, is the capacity of 3d-cards to apply more than one texture per pixel in a single-pass rendering operation. It is quite usual to find devices that can manage even four different textures simultaneously. They also allow to apply different operations to each texture channel concurrently. This operation may be a multiply or an addition, depending on the kind of lightmaps implemented. The different cases are:

1. The final result fades into black as the processed lightmap is darker. Brightest sections leave the primary texture as it is. This uses a multiply as blending operation.
2. The final result fades into white as the lightmap is lighter. So, dark sections leave the texture untouched. This involves an Add operation. This type of lightmaps usually require darker textures, that are lightened by the lightmap.

The first one is the type used in our work. It is more often used in applications because it uses normal textures (not darkened). The whole rendering process must be:

```
Set primary texture at texture unit 0 (T0)
Set lightmap at texture unit 1 (T1)
Tell T0 where to find its texture coordinates
Tell T1 where to find the GL texture coordinates
Set multiply operation between two texture units
Call to rendering function
```

8. CONCLUSIONS

Figures 8 and 9 show real-time quality compared to a picture generated by a commercial 3d rendering package. (Light attenuation with a Lambert calculation and projected shadows).

Hardware Transform & Lighting offers a noticeable speed increase on real-time environments, especially in those with a high polygon count, and we need to take advantage of that. On the other hand, lightmapping increases image quality due to the mapping of special lighting effects.

We have showed a way to solve the practical problems that appear when implementing both techniques in current 3D-cards. Our structures and methods run faster than disconnected -polygon based- lightmaps. Even more, our approach permits the creation of lightmaps without requiring manual work.

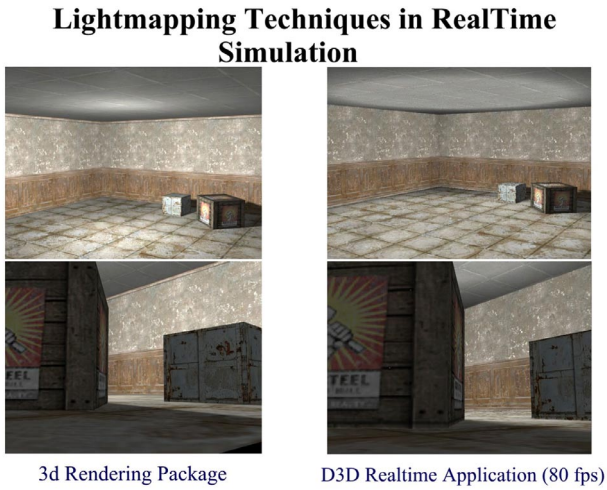


Figure 8 Realistic vs. Real-Time (I)

Figure 9 Realistic vs. Real-Time (II)

9. REFERENCES

[Abrash97] M. Abrash. Graphics Programming Black Book. Special Edition. *Coriolis Group Books* (1997), 1245 - 1271.

[Arvo] J. Arvo, D. Kirk. Fast ray tracing by ray classification. *Computer Graphics* (SIGGRAPH '87 Proceedings), 21(4), (1987), 55 - 64.

[Dietrich00] S. Dietrich. Attenuation Maps. Game Programming Gems, Ed. by M. A. DeLoura. *Charles River Media, Inc.* (2000), 543 - 548.

[Eberly99] D. H. Eberly. 3D Game Engine Design. *Morgan Kaufmann Publishers* (1999), 427 - 434.

[Möller99] T. Möller, E. Hines. Real-Time Rendering. *A K Peters, ltd.* (1999), 124 - 125.

[Thalman91] N. M. Thalmann, D. Thalmann, H. T. Minh. InterPhong Shading. Graphics Gems II. Ed. by J. Arvo. *Academic Press, Inc.* (1991), 232 - 241.

Discrete Events Simulation as a Computer Game Kernel

Inmaculada García Ramón Mollá
Computer Graphics Section
Dep. of Computation and Computer Systems
Technical University of Valencia
Camino de Vera, s/n
46022 – Valencia – Spain
{ingarcia,rmolla}@dsic.upv.es

Abstract

Many computer games follow a scheme of continuous simulation. This scheme is not able to support all possible interactions produced among different objects within the system. To join the visualization part with the simulation process is inefficient. DESK is a simulator kernel that may be used as a computer game kernel. It allows decoupling simulation and visualization, increasing calculation efficiency. Nervous sampling improves simulation accuracy, avoiding incorrect behaviors in the system object interactions, typically performed by continuous simulators.

Keywords

Hybrid simulation, computer game engine

1. INTRODUCTION

A real-time graphic application, as virtual reality (VR), computer games or simulation, may be considered a *system*, following the definition of system of Banks and Carson [Banks95]. As a system, it can be represented using *modeling* [Wainer96] and *simulation* [Banks95] techniques. Attending to the systems classification, based on the way the system evolves in time [Wainer96], a real-time graphic application could be considered as a *hybrid system*. In that, the continuous system evolution in time may be altered by events not associated to the sampling period.

The earlier real-time graphics systems worked with a main loop that coupled graphics rendering phase to simulation phase. If animation and rendering are decoupled, scenes are render more quickly even when the higher-level animation computations become complex [Shaw92]. This decoupling increases system performance [Darken95]. Some real-time graphic systems that decouples simulation and visualization are among others: the *Cognitive Coprocessor Architecture* [Roberson89], *MR toolkit* [Shaw92] based on the *Decoupled Simulation Model*, *Virtual Builder II*, [Gobbetti95], *Alice & Diver* [Pausch94] or *Bridge* [Darken95].

The rendering and simulation decoupling allows not only to increase system accuracy and speed, also allows the independence of other process in the system [Giachetti02]. The natural evolution of decoupling was to distribute the system processes in a computer network or to use parallelism. However such distribution is not possible in games created to run in a single PC or console. Network games allows multi-users, but not distribute process in the net.

1.1 Computer games

Many different source code of computer games available correspond to non-commercial free games made by enthusiastic people. These computer games lack from internal organisation. They employ rudimentary simulation techniques. That is the reason why they have not been included in the present study [ZIRON].

Only a few commercial computer games have their source code published. Among them, we emphasise DOOM v1.1 [DMCd] or QUAKE v2.3 [QKDoc] [QKCd] games and the Fly3D kernel [Watt01] (a true 3D object oriented multipurpose game kernel developed in C++) because of their importance in computer games. Their working model may be seen in Figure 1.

An events management study of different games has been performed in order to verify the use of simulation techniques. Simulation techniques applied in many cases suppose considering computer game as a continuous system, although computer games are hybrid systems. So they have large limitations when working.

2. OBJECTIVES

Hybrid Simulation (HS) allows a more accurate simulation than Continuous Simulation (CS), avoiding lost events and a disorderly execution of events when the simulator has a limit in the computing power. The objective of this paper consists in the application to video games the following issues:

- Improve simulation accuracy using a HS model.
- Verify that HS techniques can improve the system events management, increasing game efficiency.
- Decouple rendering and simulation phase, as in high-end real-time graphics applications.

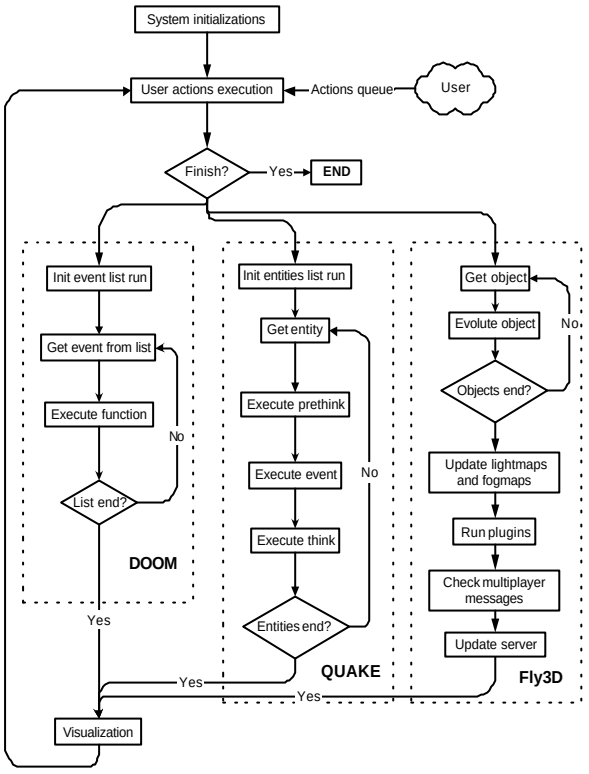


Figure 1: DOOM, QUAKE and Fly3D main loop

The saving in computer power may be used to improve the Human Computer Interface.

3. ANALYSIS

Simulation cycle is defined as the time elapsed in a run of the program main loop (sampling period of a continuous simulator). In the games analysed, each world evolution always requires a full visualisation of the entire world. All events are obligatory evolved at the higher speed that the computer system can supply, following a CS scheme. This means:

- The system is not sensitive to times lower than the sampling period.
- Events are executed in the order in which events, entities or objects management structure are accessed. They are not time ordering.
- Events are artificially synchronized matching the sampling period. They are not executed in the very moment when they happen.
- The sampling frequency depends on topics that can change during the game, as available computer power, world complexity, other active tasks in system, network overload or current simulation and visualisation load. So, the sampling frequency is variable and not predefined.
- The sampling frequency is the same for all objects, independently of their requirements. If objects behaviours do not match Nyquist-Shannon theorem, they will not be simulated properly, producing events loss, not detected collisions,... On the other hand, an object with a very slow activity may be oversampled.

3.1 Criticism

Current graphic cards support Screen Refresh Rates (SRR) over 75Hz. Higher frequencies are redundant since no flicker effect can be appreciated in practice from 72Hz on. The tests carried out upon QUAKE 3 v1.17 in current equipments employing last generation cards obtain refresh rates between 130 and 250 fps [TOM02].

This behaviour is inefficient due to the loses of time generating renderings that will never be appreciated on screen. The 70% rendering power is lost when generating 250 fps. An improvement of this operation scheme would be to separate simulation from rendering. The goal is to match the system sampling frequency to the SRR, typically 75Hz. Then, the system calculates only as many frames as the number of times the screen is refreshed. Released computer power provides a bigger amount of simulation cycles used to calculate simulations more precisely [Reynolds00]. Decoupling the system is not the solution if the simulation scheme remains continuous, because:

- Simulating at the fastest speed is inefficient since it spends calculations simulating intermediate states not rendered on screen (unadjusted sampling period).
- Scene graph must be accessed twice.

A common practice in the earlier computer games, with low scene complexity, was to simulate and render each object, accessing the scene graph once [Bishop98]. Current situation is different because rendered frame rates are higher than SRR [TOM02].

On the other hand, access through the scene graph when many objects will never generate events, is quite inefficient. It would be better to use an events queue. Only those objects that generate events will be checked, avoiding to access the remainder objects.

3.2 Improvements

After analysing how these programs operate, the following improvements are proposed:

- All the objects will have the same priority, including player avatar.
- Nervous sampling will be supported. This means:
  - No event may be lost.
  - Events should be attended and simulated at the moment in which they are produced, not in the following cycle.
  - Each object may have its own sampling period. Therefore, those objects generate events at a higher rate have to be sampled at a higher frequency. Also, the objects that do not generate events, do not overload the simulation engine.
  - Events should be simulated ordering in time.

4. DISCRETE EVENTS SIMULATION KERNEL

DESK [Garcia00] is a C++ object oriented Discrete Events Simulation Kernel API. This simulator allows changing dynamically system topology. Also, it allows include any continuous or discrete behavior in the system. It uses to

be faster than Smpl [SMPL] for almost all simulated models. DESK can manage models with high complexity and size. The programmer only focuses on the model description, not in events or programming bugs. The effort to describe a behavior can be reused (in the same model or others). The programmer can implement the definition of the behavior functions. This gives flexibility enough to use it as a game kernel.

5. USING DESK AS A COMPUTER GAME KERNEL

Using DESK as a simulation engine for computer games does not imply to change topics as the structure of the scenes description files or characters. It does not modify the file parser, the scene graph, the rendering techniques applied or video game style. It only modifies the system events management and introduces a HS scheme.

Objects in DESK, interact by *messages passing*. A message is modeled by means of a client that contains the necessary information to develop an adequate interaction. For example, when a projectile collides with a wall, it sends a message to the wall indicating that it has collided with it. The message must includes the necessary parameters (like mass, point of impact, velocity or projectile kind). The behavior of the receiving object depends on the object transmitter and the kind of message sent. The receiving object determines what kind of behavior must be presented. For example, if a small stone collides with a wall, perhaps the wall attributes will not be affected with that message (depending on the weight of the stone). If the projectile is a missile, the wall can disappear. The same event with different transmitting objects causes different behaviors in the same receiving object; but also in the transmitter (the bullet rebounds if the wall is made of concrete and it is incrusted if is a clay wall). Different kind of objects will have different behaviors.

On the other hand, objects spent time generating an answer to the message. Therefore, when a ball strikes a wall, it does not rebound immediately. It spent time changing its trajectory. The ball can be deformed as consequence of the impact, to subsequently recuperate its original form.

5.1 Creating a Game

When a game starts, all objects in game must be created, assigning them a behavior and grouping them into a hierarchy. The simulator will invoke the behavior assigned to an object at the precise moment. It is possible to define autonomous system behaviors.

The system generates a render event for each SRR. User events are registered in the events queue at the moment that are produced. So, they are mixed with the remainder of system events, being ordered in time. In this way, the player receives the same homogeneous processing that the remainder entities. Computer game implementation using DESK supposes:

- To define the behavior methods of each object:
  - A behavior differentiated depending on the kind of message received and the transmitter object.
  - Generation of new events produced as consequence of the defined behavior. It can include answers to transmitter object.
  - Statistical methods, if proceeds.
- To define the system behavior. This behavior should include, al least, the SRR.
- To invoke the simulator initialization function, in order to initialize simulator internal data structures.
- To start the simulation.
- To obtain results as statistics or accounting.

5.2 Working Model

When an animated object is created, a new event is sent to the system. Unanimated objects will not generate any event at the moment of creation. When the entire scene is created, the computer game starts.

Scene simulation process is carried out independently of the visualization process. During computer game execution, the system generates an event for each screen refresh. When events manager detects the event, two things may happen:

- Exist already a screen previously calculated which is waiting to be shown on screen. Therefore, there is no reason to process a new render.
- The last calculated image is being shown on screen, there is no screen available waiting to be shown. So, a new rendering must be calculated.

The visualization freezes the simulation while it generates the new image going through the scene graph. Therefore, there is only necessary to go through the scene graph once for each visualization as in typical games.

Each time a screen refresh ends, the graphic package activates the screen calculation. So, next time a refresh event is generated, will be carried out the visualization.

When the calculated frame rate is lower than the SRR, DESK behavior will be equivalent to the studied games. If the calculation power is high, computer will not lose time calculating an image that will never be rendered. This remainder computer power may be dedicated to improve the simulation accuracy or other needs, as opposed to these games (see point 3.1).

For example, let it be a hand grenade thrown to the air. At the moment in which it is thrown, an event of blowing is generated in 1.76 seconds, wherever it is. While it does not blow, the hand grenade travels through a continuous trajectory by air. Trajectory is sampled in screen SRR times by second. Therefore, the hand grenade position must be recalculated for each screen. That is done generating an event each 1/SRR seconds. It is not necessary to spend computer power simulating intermediate states, since only the last one is shown at screen. DESK simulation model is hybrid because it carries out a CS, sampling system each 1/SRR seconds, while it still bears discrete events.

DESK always performs SRR fps. So, sampling frequency in DESK is constant assuming that there is computer

power enough. It does not depend on the game load or computer power available.

The HS model bears implicitly nervous sampling because those processes that present a great variability will generate a great quantity of events consuming more computer power. Those objects that do not generate events do not overload the game. So, computer power is distributed among all objects depending on their behaviors.

5.3 Results

Coupled and decoupled models have been simulated in the laboratory. Let it be: T_S the time used by the videogame to simulate a game step, T_R the time spent to render a frame, and, finally, ν_V the fps actually generated by the videogame. The conclusion for CS decoupled systems are:

1. $SRR \geq \nu_V$
2. If $T_S+T_R \leq 1/SRR$ then $SRR=\nu_V$
3. If $T_S+T_R > 1/SRR$ then $T_S+T_R \leq 1/\nu_V$
4. The amount of simulations is always equal or higher than the amount of renderings

Point 2 gives the maximum performance because SRR is achieved. Videogames behaves better in the point 2 situation than point 3. Using DESK HS reduces T_S, increasing the probability to operate under the conditions of point 2.

6. CONCLUSION

The computer games analyzed follow an inefficient simulation scheme because of they use CS and they do not decouple visualization from simulation. Although CS model is widely used in computer games and it works in practice, this model becomes insufficient to simulate accurately more complex behaviors like next generation computer games, VR applications or interactive real time graphics. HS allows a more accurate simulation than CS, avoiding lost events and a disorderly execution of events when the simulator has a limit in the computing power.

When DESK is used as a computer game kernel, it allows HS and forces to decouple the system. So, the calculation efficiency is increased and it improves the simulation accuracy performing a better final result. Visualization and simulation decoupling allows using parallel techniques in order to increase computer game speed, to bear more complexes computer games, professional applications or to improve final visual result.

Since DESK supports nervous sampling, simulation accuracy is improved, avoiding incorrect behavior to be produced. Computer power is concentrated in simulating those parts of the game with high rate of events generation, dedicating less attention to those parts with a lower events rate.

7. ACKNOWLEDGEMENTS

This work has been partially funded by the Spanish government CICYT TIC1999-0510-C02-01

8. REFERENCES

[Banks95] J. Banks, J.S. Carson, B.L. Nelson. Discrete-Event System Simulation. (Prentice Hall, 1995)

[Bishop98] L. Bishop, D. Eberly, T. Whitted. Designing a PC Game Engine. IEEE CG&A, vol. 18, no. 1, January/February 1998, pp. 46-53

[Darken95] R. Darken, C. Tonnesen, K. Passarella. The Bridge Between Developers and Virtual Environments: a Robust Virtual Environment System Architecture. Proceedings of SPIE 1995, No. 2409-30

[DMCd] www.idsoftware.com/archives/doomarc.html

[Fishman78] G.S. Fishman. Conceptos y Métodos en la Simulación Digital de Eventos Discretos. (Limusa, 1978)

[Garcia00] I. García, R. Mollá, E. Ramos, M. Fernández D.E.S.K.: Discrete Events Simulation Kernel, ECCOMAS 2000, September 2000

[Giachetti02] M. Agus, A. Giachetti, E. Gobbetti, G. Zanetti. A Multiprocessor Decoupled System for the Simulation of Temporal Bone Surgery. *Computing and Visualization in Science*, To appear

[Gobbetti95] E. Gobbetti, J.F. Balaguer. An Integrated Environment to Visually Construct 3D Animations. SIGGRAPH 95 Conference Proceedings, Annual Conference Series, pp 395-398, August 1995. ACM SIGGRAPH, Addison-Wesley.

[Pausch94] R. Pausch et al. Alice & DIVER: A Software Architecture for the Rapid Prototyping of Virtual Environments. Course notes for SIGGRAPH '94 course, "Programming Virtual Worlds".

[QKCd] www.quake2.com/kko/

[QKDoc] www.gamers.org/dEngine/quake/

[Reynolds00] C. Reynolds. Interaction with Groups of Autonomous Characters. Game Developers Conference Proceedings, March 2000.

[Roberson89] G.G. Robertson, S.K. Card, J.D. Mackinlay. The Cognitive Coprocessor Architecture for Interactive User Interface, UIST'89 Proceedings, 1989, pp. 10-18.

[Shaw92] C. Shaw, J. Liang, M. Green, Y. Sun The Decoupled Simulation Model for Virtual Reality Systems. CHI'92, May 1992, pp. 321-328.

[SMPL] http://www.autoctrl.rug.ac.be/ftp/smpl/

[TOM02] www6.tomshardware.com/graphic/02q1/020304/geforce4-09.html

[Wainer96] G.A. Wainer. Introducción a la Simulación de Sistemas de Eventos Discretos. Technical Report: 96-005. Buenos Aires University.

[Watt01] A. Watt, F. Policarpo, 3D Games. Real-time Rendering and Software Technology, (Addison-Wesley, 2001) http://www.fly3d.com.br

[ZIRON] http://www.ziron.com/links/

Displaced $\sqrt{3}$ Subdivision Surfaces

Muhammad Hussain, Yoshihiro Okada, and Koichi Niijima
Graduate School of Information Science and Electrical Engineering, Kyushu University,
6-1, Kasuga Koen, Kasuga, Fukuoka 816-8580, Japan.
mhussain@i.kyushu-u.ac.jp

Abstract

Displaced subdivision surface representation has a number of benefits such as editing, geometry compression, animation, scalability, and adaptive rendering. This surface representation encodes a detailed surface model as a scalar-valued function defined over a smooth domain surface. The defining of smooth domain surface is the challenging task in this representation. In this paper we introduce a new method to define smooth domain surface based on $\sqrt{3}$ subdivision. Our proposed algorithm is computationally more efficient and consumes less memory as compared to the original algorithm by Lee et al.[Lee00] and the resulting surface has more levels of detail due to the specific nature of $\sqrt{3}$ subdivision if a prescribed target complexity of the mesh must not be exceeded.

Keywords

Meshes, subdivision, multiresolution, displacement map, geometry compression, irregular connectivity

1. INTRODUCTION

The emergence of latest technological systems ranging from CAD systems to range sensing and isosurface extraction has given rise to highly detailed and complex surface models. Representing these models with triangle meshes has become a de facto standard. Models consisting of millions of triangles are becoming commonplace. Geometry of a model is typically encoded by three scalar values (x, y, z) for each vertex of a representative mesh and the connectivity of the mesh is often irregular. Sheer size and irregular connectivity of such meshes put a threat to manipulation, transmission, storage, animation and rendering of surface models.

An alternative approach is to represent a surface model as a displacement field over a simple, smooth domain surface. Representing an arbitrary surface as a displacement field puts the challenging problem of finding a smooth underlying domain surface. Lee et al [Lee00] define domain surface using subdivision surfaces due to their capability of representing smooth surfaces of arbitrary topology. Displaced subdivision surface representation consists of a control mesh and a scalar function which displaces the associated domain surface locally along the normal. Lee et al [Lee00] have shown in their original work that this representation offers a number of advantages over the mesh representation of a surface model. We introduce a new method to define domain surface using $\sqrt{3}$ subdivision technique [Kob00]. Construction of control mesh from the mesh representing the original surface is the basic problem in defining domain surface. Lee et al use a very slow and memory consuming method to simplify the mesh followed by a time consuming process to optimize the vertex positions. We construct the control mesh using a fast and memory efficient technique to simplify the original mesh and, a fast and rigorous method to optimize the vertex positions. Our algorithm is similar to that of Lee et al. [Lee00] except the method of construction of domain surface.

The idea of displacing a surface by a function was first introduced by Cook[Coo84]. Guskov et al.[Gus00] present an algorithm to represent a surface as a normal mesh by successively applying a hierarchy of displacements to a mesh as it is subdivided. Their construction encodes most part of the mesh as scalar displacements, a small fraction of vertices need vector displacements to prevent surface folding. Recently Won and Chang [WC01] presented an algorithm that directly reconstructs displaced subdivision surface from unorganized points. This method is fairly efficient but it is limited only to a small class of surfaces having simple topology.

The subsequent paper is arranged as follows. Section 2 briefs $\sqrt{3}$ subdivision surface and outline of our algorithm has been presented in Section 3. In Section 4, we elaborate our algorithms to define smooth domain surface. Results have been discussed in Section 5. Section 6 concludes the paper.

2. $\sqrt{3}$ SUBDIVISION

Among subdivision schemes published in the literature which perform on triangular meshes, 4-8 subdivision [Vel00] and $\sqrt{3}$ subdivision [Kob00] are based on 1-to-2 and 1-to-3 splits respectively and as such are slower than Loop scheme [Loo87] in the sense that they increase the number of vertices (and faces)

by the factor of 2 and 3 respectively. As a consequence, using 4-8 subdivision and $\sqrt{3}$ subdivision, we have more levels of uniform resolution if a prescribed target complexity of the mesh must not be exceeded. For 4-8 subdivision, the position mask for infinite position of a vertex is not straight forward, so we have decided to use $\sqrt{3}$ subdivision for our displaced subdivision surface.

For $\sqrt{3}$ subdivision, the topological operation performs a 1-to-3 split for every triangle by inserting the new vertex at its center and introducing three new edges connecting the new vertex to the surrounding old ones. In order to re-balance the valence of the mesh vertices, every old edge that connects two old vertices is flipped. Smoothing operator consists of two rules: one for odd vertices and one for even vertices. For further detail consult [Kob00]. Using the technique described in [Sta8], the tangent masks are calculated to be

$$(c_0, c_1, ..., c_{n-1}) \text{ and } (c_{n-1}, c_0, ..., c_{n-2})$$

with $c_i = \cos(2\pi i/n)$.

3. OVERVIEW OF THE ALGORITHM

Our algorithm takes triangle mesh as an input and outputs control mesh of the subdivision surface along with displacement field values. It performs the following steps to convert a given triangle mesh representation of a surface into displaced $\sqrt{3}$ subdivision surface.

1. Simplification: The original mesh is simplified to obtain the raw control mesh.
2. Optimization: The vertices of the raw control mesh are optimized so that the domain surface closely fits the original surface.
3. Subdivision and Sampling: The difference between the original surface and the domain surface is simplified as displacement field values.

Figure 1 illustrates the conversion process.

The main achievement of our algorithm is memory consumption and computational efficiency. We simplify the given triangle mesh using a fast and memory efficient technique to simplify the mesh. To optimize the vertex positions, we use a simple method which does not involve energy minimization, but the solution of a simple consistent sparse linear system.

4. MAIN COMPONENTS

4.1. Simplification

We exploit a fast, simple and memory efficient error metric [Hus01] and a sequence of half-edge collapse transformations to simplify the triangle mesh to obtain the control mesh for subdivision surface. Consider an arbitrary half-edge collapse transformation $u \rightarrow v$, as shown in Figure 2(a). The vertex u and the two triangles incident on the edge $e = \{u, v\}$ will be eliminated and the remaining triangles incident on u replace u with v. The cost of this transformation is

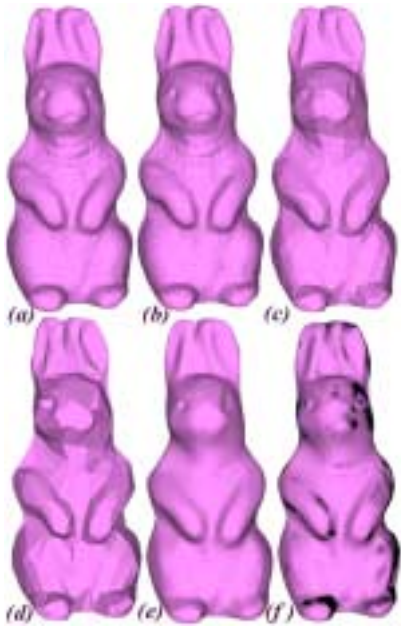


Figure 1: (a) Original rabbit model (Courtesy Cyberware) T:134074, V:67039. (b) Displaced $\sqrt{3}$ subdivision surface, T:64476, V:32240. (c) Initial control mesh, T:796, V:400. (d) Optimized control mesh. (e) Smooth domain surface, k = 4. (f) Displacement field.

$$\text{Cost}(u \rightarrow v) = \sum_{e \in E} \theta_e A_e$$

where $E = \{e_0, e_1, e_2, e_3, e_4\}$ and A_{e_i} is the area of the triangle $t_i'' = \{u, v, v_1\}$ swept out by the edge e_i as it moves when edge e is collapsed and θ_{e_i} is the angle between the normals of the adjacent triangles t_i and t_i'before and after the transformation, see Figure 2(b). For the sake of efficiency, θ_{e_i} is approximated by $1 - n_{t_i} \cdot n_{t_i^0}$.

To ensure that the normal space of the original surface is locally preserved, we restrict the edge collapse transformation $u \rightarrow v$ which result in the normal at the vertex v that deviates significantly from the original normal at vertex v. We compute normal at each vertex of the original triangle mesh as a preprocessing and store it for comparison.

4.2. Optimization of the Control Mesh

After control mesh have been constructed , we optimize the vertex positions such that the resulting subdivision domain surface closely fits the original surface. To achieve this objective we introduce a very simple and rigorous technique based on $\sqrt{3}$ subdivision rule for the position of a vertex at subdivision level m which has been described in [Kob00]. In particular for $m = 1$, this rule can be expressed as

$$p^1 = \gamma_n p + (1 - \gamma_n) p^\infty$$

with $\gamma_n = (\frac{2}{3} - \alpha_n)$. But according to the smoothing rule for even vertices [Kob00]

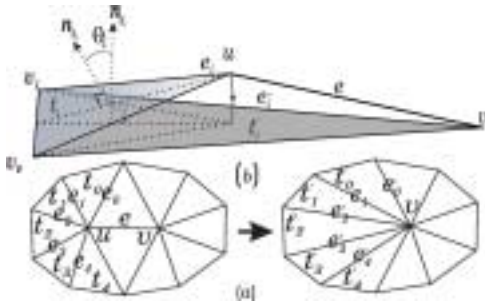


Figure 2: (a) Edge collapse operation (b) Collapse of edge $e = \{u, v\}$ will map the edge e_i onto e_i' and the adjacent triangle t_i onto t_i'.

$$p^1 = (1 - \alpha_n)p + \frac{\alpha_n}{n} \sum_{i=0}^{n-1} p_i.$$

From these two relations, we obtain the following vector equation

$$\frac{1}{3}p + \frac{\alpha_n}{n} \sum_{i=0}^{n-1} p_i = (1 - \gamma_n)p^\infty.$$

Here p^∞ is the position of the vertex p on the limit subdivision surface and p_i 1-ring neighbors of p. We require that the vertices of the raw control mesh must be on the limit surface, so we replace p^∞ in the above equation with the corresponding vertex position on the control mesh because the vertices of the raw control mesh, which is the result of the application of a sequence of half-edge collapse transformations, are on the original surface. Likewise, we obtain a vector equation corresponding to every vertex on the raw control mesh. These equations constitute a sparse linear system; solving this system for each of the $x-$, $y-$, and $z-$components, we get the optimized positions of the vertices which will lead to domain surface that closely fits the original surface. This system can be solved in linear time using conjugate gradient method.

To enhance the quality of optimized control mesh, we smooth it slightly using tangential component of Laplacian smoothing operator [Tau95]; it will improve the aspect ratio of triangles and will not have shrinkage effects. Discrete form of the Laplacian operator L is

$$Lp_i = \frac{1}{n} \sum_{j \in N(i)} p_j - p_i$$

with n being the valence of p_i and $N(i)$ the 1-ring neighborhood of p_i. Its tangential component L_t is given by

$$L_t p_i = Lp_i - (Lp_i \cdot \mathsf{n})\mathsf{n}$$

where n is unit normal at vertex p_i. The relaxed position of vertex p_i is expressed as follows

$$p_i' = p_i + \lambda L_t p_i$$

where λ is a parameter to control the effect of tangential Laplacian operator; $\lambda = 1$ results in uniformly sampled mesh but at the loss of essential features. The best choice for the value of λ is 0.2 [WC01].

Model	Original Size	D.M. Size	C.M.Size	% L^2
Horse	96966	64476	796	0.04
Rabbit	134074	64476	796	0.017
Venus	268686	80676	996	0.012
Balljoint	274120	85536	1056	0.02

Table 1: Sizes of the original model, corresponding displaced mesh (D.M.) and control mesh (C.M.) are given as the number of triangle faces.

Process	Horse	Rabbit	Venus	Balljoint
Simplification	16.69	23.30	49.93	51.30
Optimization	0.15	0.15	0.20	0.26
Sampling	14.05	14.88	27.89	28.28
Total (sec)	30.89	38.33	78.02	79.84

Table 2: Time of conversion in seconds.

4.3. Subdivision and Sampling

Optimized control mesh is subdivided k times and the resulting mesh is then pushed to its limit position using $\sqrt{3}$ subdivision rules described in [Kob00]. At each vertex of the subdivided mesh, we compute the limit position and normal of the domain surface. To capture the detail of the original surface as a displacement map over the smooth domain surface, we compute the signed distance from each vertex of the smooth domain surface to the original surface along the normal. The position of a vertex on the smooth domain surface and the normal thereat define a straight line. The intersection of this line with the original surface provides the signed distance. To compute the intersection of this line efficiently we store the original triangle mesh in a OBB tree data structure [GLM96]. This line may have multiple intersections or the original surface may be oriented in the wrong direction with respect to this line. If the directed line is intersected at more than one points, then we pick the one closest to the domain surface. In the second case we reject the intersection.

5. RESULTS AND DUSCUSSION

Using our implementation of the algorithm described in the preceding section, we converted many closed triangle meshes into displaced $\sqrt{3}$ subdivision and found encouraging results. Some triangle meshes together with displaced $\sqrt{3}$ subdivision surfaces have been shown in Figures 3. After getting control meshes optimized, we subdivide each model up to subdivision level 4 and then push the vertices to their infinite positions by applying position mask.

To evaluate the surfaces generated by our algorithm, we use I.E.I-CNR Metro tool. Table 1 shows the mean square error L^2 as the percentage of the diagonal of bounding box. It is obvious that the surfaces generated by our method compare well with the original ones.

The times of conversion have been shown in Table 2. These times have been reported on a 550MHz Pentium III machine. The execution times of the original displaced subdivision sur-

Process	armadillo	venus	bunny	dinosaur
Size(#F)	210,944	100,000	69,451	342,138
Simplification	61	28	19	115
Optimization	25	11	11	43
Sampling	2	2	1	5
Total (min)	88	41	31	163

Table 3: Time of conversion in minutes [Lee00].

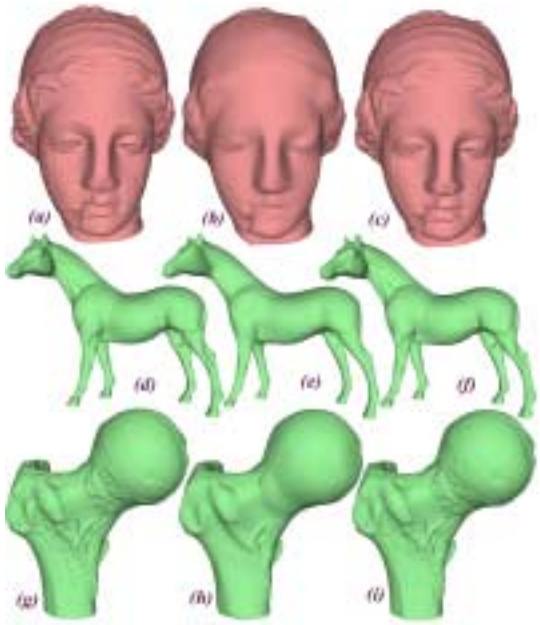


Figure 3: *Venus model*: (a) Original T: 268686, V:134345. (b) Smooth domain surface, k = 4, T: 80676, V:40340 (c) Displaced $\sqrt{3}$ subdivision surface. *Horse model*: (d) Original , T: 96966, V:48158. (e) Smooth domain surface, k = 4, T:64476, V:32240 (f) Displaced $\sqrt{3}$ subdivision surface**.** *Balljoint model*: (g) Original T: 274120, V:137062. (h) Smooth domain surface, k = 4, T:85536, V:42770. (i) Displaced $\sqrt{3}$ subdivision surface.

face scheme, shown in Table 3 [Lee00], are also have been obtained on 550 MHz Pentium III PC. Although in our experiments, we use different models, even then we can have an overall idea about the efficiency. From the times of conversion reported in Tables 2 and 3, we can safely conclude that our approach to define smooth domain surface is much more faster.

Lee et al. [Lee00] define parametrization of the original surface to keep track of the original geometry while simplifying the mesh. Moreover they use Quadric error metric [GH97] with heuristics, which is not memory efficient, to simplify the mesh. That is why our algorithm is fast and memory efficient as compared to the one by Lee et al.

6. CONCLUSION

In this paper, we have presented a new algorithm to define smooth domain surface for displaced $\sqrt{3}$ subdivision surface, which is fast and memory efficient. Although we have given the details of our algorithm keeping in view $\sqrt{3}$ subdivision due to its special characteristics, this can easily be implemented with other subdivision schemes. As our main contribution is the efficient method to construct smooth domain surface for a displaced subdivision surface, so our algorithm offers all those benefits as the one by Lee et al., i.e. compression, editing, animation, and scalability. This algorithm has been initially implemented for closed surfaces, its extension to open surfaces is still under investigation.

References

[Coo84] R. Cook. Shade Trees. In *Computer Graphics* (*Proc. SIGGRAPH'84*), 18(3), pages 223-231, 1984.

[GH97] M. Garland, P. Heckbert. Surface Simplification using Quadric Error Metrics. In *Computer Graphics* (*Proc. SIGGRAPH'97*), pages 209-216, 1997.

[GLM96] S. Gottschalk, M. Lin and D. Manocha. OBB-tree: A Hierarchical Structure for Rapid Interference Detection. In *Computer Graphics* (*Proc. SIGGRAPH'96*), pages 171-180, 1996.

[Gus00] I. Guskov, K. Vidimce, W. Sweldens and P. Schroder. Normal Meshes. In *Computer Graphics* (*Proc. SIGGRAPH'00*), 1995.

[Hus01] Hussain, M., Okada, Y. and Niijima, K. : Fast, Simple and Memory Efficient Mesh Simplification. In *Proc. Fourth IASTED International Conference on Computer Graphics and Imaging* (*CGIM2001*), pages 72-77, Hawaii USA, August 2001.

[Kob00] L. Kobbelt. $\sqrt{3}$ Subdivision. In *Computer Graphics* (*Proc. SIGGRAPH '00*), August 1996.

[Lee00] A. Lee, H. Moreton, and H. Hoppe. Displaced Subdivision Surfaces. In *Computer Graphics* (*Proc. SIGGRAPH '00*), pages 85-94, August 2000.

[Loo87] C. Loop. Smooth Subdivision Surfaces based on Triangles. Master's thesis, University of Utah, Department of Mathematics, 1987.

[Vel00] Luiz Velho,and Denis Zorin. 4-8 Subdivision. *CAGD*, volume 18, Issue 5, Pages 397-427.

[Sta8] J.Stam. Exact Evaluation of Catmull-Clark Subdivision Surfaces at Arbitrary Parameter Values. In *Computer Graphics* (*Proc. SIGGRAPH'98*), pages 395-404, July 1998.

[Tau95] G. Taubin. A Signal Processing Approach to Fair Surface Design. In *Computer Graphics* (*Proc. SIGGRAPH'95*), pages 351-358, August 1995.

[WC01] Won K.J. and Chang Hun Kim. Direct Reconstruction of Displaced Subdivision Surface from Unorganized Points. In *Proc. Pacific Graphics*'01, pages 160-168, 2001, Tokyo Japan.

Empty Museum. An Inquiry on Autonomous VR Systems and Hybrid Spaces.

Luis A. Hernández Javier Taibo Antonio Seoane

Visualization for Engineering, Architecture and Urban Design Group
University of Coruña – Spain
lahernandez@iccp.udc.es,jtaibo@udc.es,ynot@udc.es

Abstract

This paper describes a new system for experimentation of three-dimensional, virtual reality (VR) environments, based on the creation of an immersive and walkable space we have named Empty Museum. The user can walk physically and freely around this virtual, three-dimensional space, without any wire connections and while observing multimedia contents included into the same area. Work has consisted on the design of the hardware and controlling software for this immersive space, but also on the creation of an application that allows the user to visualize any type of VRML 2.0 content in this VR hall, enabling us to create various animated and interactive worlds. These experimental worlds have served, on one hand, to analyze the user's response to new forms of interaction occurring in the virtual, walkable space, and on the other, to discern new forms of creating contents that can derive from the use of real space as part of the actual interface.

Keywords

Virtual Reality, Virtual Worlds, Wireless, Wearable Computing, Mobile Computing, Direct Interaction, Interaction Techniques.

1. INTRODUCTION

In recent years, the term "immersive" applied to experimentation with virtual spaces has acquired the meaning of "surrounding" in the sense of "all that surrounds the spectator". For a virtual setting to be considered as such, the user must, at all time, find the expected image that would correspond to the virtual surrounding environment, regardless of the direction looked at.

Until now, "immersion" in an environment of these characteristics has not gone much further. The sensation of three-dimensionality could be increased with the use of stereoscopic image and spatialized sound. However, the natural way of experimenting space as humans know it, i.e. observing while moving through it in any direction, made it necessary to use hardly believable metaphors [Bowman98]. That is, putting into the user's hands a joystick, mouse or other similar type of interface that enabled a control of direction, and asking the person to pretend to be driving around a virtual setting in an imaginary car or by means of other external forms of propulsion.

The necessary use of the imagination to perceive the metaphor of the vehicle as something real takes away all naturalness and, consequently, much of the reality to the experience of the exploration. To experiment a space in the real world, it is necessary to move through it, understanding by movement not only the motion in itself, but also the kinaesthetic sensation of perceiving one's own body in movement, given that the interpretation of one's own movement facilitates the comprehension of the scale of what surrounds us.

The action of exploring requires a personal rhythm of movement, which differentiates it from the action of travelling. In the latter, the objective is to arrive to a certain point and any variations in the movement are secondary to the perception of the actual journey in itself and, therefore, to the finality of the movement. Yet, in the exploration of space, i.e. when observing a cathedral or a museum, it is necessary for the spectator to decide the rhythm of the movement, the pauses, the accelerations, when to move faster or slower, when to browse or move sidewards, turn or go back. The simulation of all the possible forms of movement by using a metaphor and a device such as a joystick is very difficult, specially considering that movement is usually limited to two modes, i.e. standstill and forward at a constant speed, generally with an abrupt and non gradual transit from one state to the other.

The reason for using the metaphor of the vehicle is no other than the dependence on the physical location of the computer that generates the images, that remains static

while simulating the movement. Until now, the complexity of the process of visualizing three-dimensional environments required the use of computers with high graphics performance, which were not very movable. This limitation has been overcome with the arrival of portable computers with high graphics performance. Another technological constraint was occasioned by the volume, weight and power supply needs of the HMD (Head Mounted Display). This is no longer a problem since the existence of small, light-weight, battery operated systems, i.e. Glasstron or similar. Another obstacle to be overcome was the restriction to movement occasioned by the cables that connected the user to the 6 degrees of freedom (DOF) tracking system. This tracking system has a fixed element that acts as a position reference and cannot move around with the user. However, in the market there are now some alternatives i.e. Intersense IS600 Mark2 [Foxlin98] that allow you to obtain the position in space of a small, light-weight, wireless and battery-operated sensor. This makes it possible to finally relieve the user from the eternal umbilical chord that traditionally connected him or her to the VR engine.

We found various systems now using this new approach, with different target applications, i.e. MARS by the Columbia University [Höllerer99] [Feiner97], Cybercompanion [Cybercompanion] or Archeoguide [Didier01].

The idea is to use real space as one of the elements in the motion interface and as part of the actual virtual space. This, without doubt, facilitates the experimentation of synthetic, architectonic-type spaces. However, the fundamental contribution of this new approach is to open the way to the use of a new form of contents, where the experience of the visit goes further than simple contemplation. Although there is still much to do, we have started to study the parameters associated to the perception of one's own movement and worked on new forms of interaction with the virtual contents, based on the location of the observer.

2. SYSTEM DESIGN

2.1. Hardware Architecture

For the physical construction of the VR system we sought for commercial "off the shelf" elements that could be integrated satisfactorily to meet the requirements observed. This allowed us to build an acceptably light and comfortable system in a short time and with low costs.

The final design of the first Empty Museum prototype placed the rendering computer in the mobile unit carried by the user, and communication with the tracking system was conducted through a wireless network. The diagram of Figure 1 depicts the components and connections between them. This system provides the user with an autonomous system (not limited by the presence of the traditional umbilical cable connected to the VR engine) so he or she can walk freely in a space of up to 8x8 metres, sufficient, as we shall see, to display spaces which are even larger.

2.2. Software Design

The system's software architecture is composed of two applications, corresponding to the fixed-mobile division explained in the previous section (see Figure 1): a controlling application that is run on the fixed PC, that manages the wireless tracking system (*base*), and a visualization application that is run on the laptop carried by the user (*satellite*).

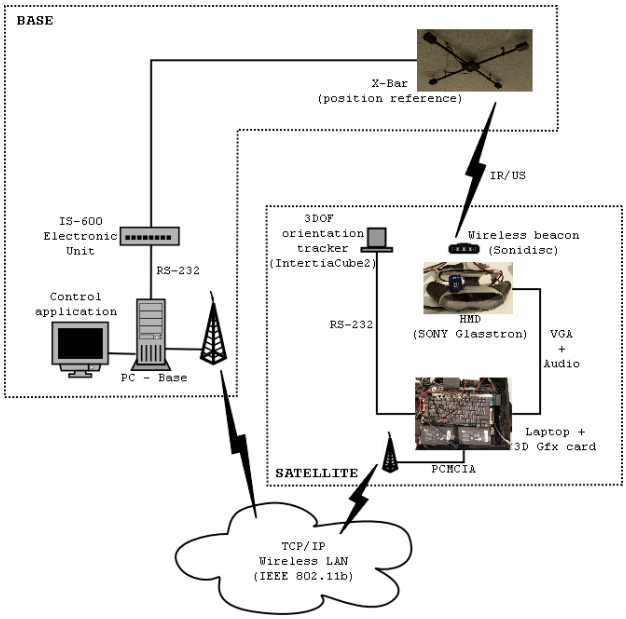


Figure 1. Hardware Architecture

The base application monitors the connections of the satellites present in the system and allows the operator to view the condition of these satellites, disconnect them, calibrate their tracking devices, assign virtual worlds to them, etc.

The satellite application continuously receives data of its position from the base through the wireless network and of its orientation from the gyroscopic/inertial/magnetic system. From this data, it generates, in every moment, the image corresponding to the position of the user's head. This image is displayed on the HMD screen.

For the construction of the contents of the virtual worlds we adopted the VRML 2.0 format, for its versatility, that, apart from defining the scenegraph with geometry, materials, textures, lighting, etc., permits the incorporation of spatialized 3D sound, animations, videos and behaviours controlled by sensors of proximity, touch, visibility, etc. In general, it provides a great potential for interaction and combination of multimedia elements.

Using this format also makes it possible to build worlds with authoring tools i.e. CosmoWorlds. To interpret the VRML 2.0 information, a viewer was developed based on the kernel of the open source library OpenVRML

[OpenVRML], which was adapted to this specific application and new features were added.

The software was written in C++ language and OpenGL as the API for the graphics hardware. Spatialized sound was implemented using the OpenAL library [OpenAL]. For the development of the controlling application's interface, we used the GTK+ library [GTK].

Even when the actual implementation of the system uses a specific hardware, the software has been designed and developed in a modular way and is thus completely configurable. It could be adapted to use combinations of different devices without making changes in the programs. The configuration files used are defined in a XML language and we developed graphical tools to generate them. The software architecture of the system is illustrated in Figure 2.

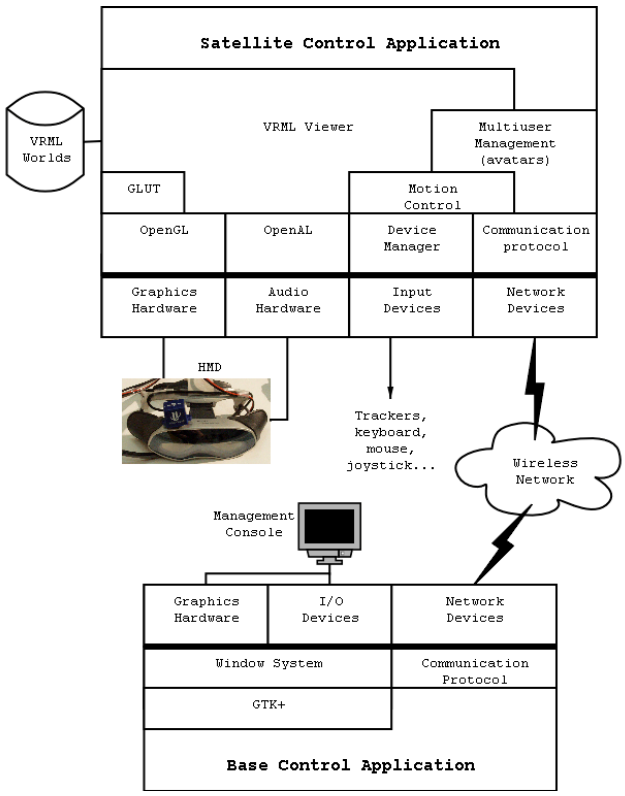


Figure 2. Software architecture.

The software was developed so that various users could visit the same or different virtual worlds at the same time, in the same real/virtual space. The users can optionally see each other as an avatar, which is a VRML model included in the definition of the virtual world and which moves associated to the corresponding user's position. The protocol used in the communications system is based on TCP/IP.

3. HYBRID SPACE

The user of the Empty Museum experiments space in a double manner. On one hand, the space as what he or she knows as such, with known dimensions, in a real area the person is conscience of being in. On the other hand, in that same space there are virtual objects. Not only does the user see and accept these as inserted in the space, as happens in the augmented reality systems, but is also able to experiment their size in relation to him or herself through parallax and movement around, nearer and away from them. The actual virtual space (not the objects inserted in it) is, therefore, as genuine as the actual real space, inasmuch as it has its same properties. As a user, you see yourself immerse in a hybrid space in which it is easy to move around and observe the objects. The real space thus becomes part of the interface.

The dimensions of both spaces can, however, be different in size, and this difference can be manipulated by introducing a scale factor in the movement. This way, we can lodge a large virtual space into a much smaller dimensioned real space. The user moves around, taking steps that can cover various metres of the virtual space, giving a sensation of very fast movement that is surprisingly not unpleasant. On the contrary, the testers described it as natural and fluent. This allows us to display great virtual spaces in a small room. When experimenting with an example of architectonic space, we displayed the inside of a building more than 30m long in an Empty Museum of only 8m long. For worlds that are much larger or complex, we have also experimented with the "teleport chamber" metaphor.

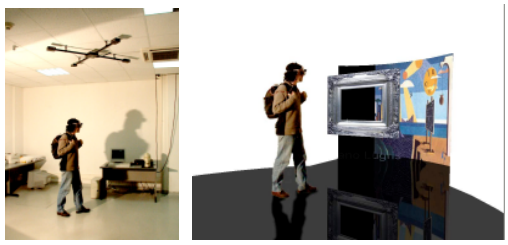


Figure 3. Real Space and Virtual Space

4. PERCEPTION IN THE WALKABLE VIRTUAL SPACE

We can define reality as the cognitive interpretation of what we perceive, that is, the interpretation of the information an individual receives from the surrounding environment through his or her senses. Therefore, to make a person feel immerse in a reality that is different from the real, physical world, you must fill that person's senses with information of the virtual world you want to introduce him or her into. If you only stimulate some of the senses in this direction, the user will be receiving information from two different environments simultaneously, allowing him or her to distinguish which one is real and which virtual. The more senses you artificially stimulate, the more the user becomes immersed in the virtual world.

The major difference between this and previously described systems of virtual space experimentation is the extraordinary contribution that real movement inside the

room has to the sense of realism of the virtual space. This is because, when you move, your own kinaesthetic and vestibular systems provide you with information that corresponds to what you are seeing, and to this we add the traditional stimulation of visual and auditory senses.

During the testing and evaluation of the Empty Museum we did a poll to several users with different skill levels. This poll confirmed that the sense of immersion achieved was greater than that obtained by any other traditional VR system, including CAVE.

This sensation can be observed from the outside, through the expressions and gestures of the user who is exploring the worlds. It is frequent to see the person moving to avoid obstacles, even though these are virtual and do not present any physical adversity or obstruction, i.e. the pillars of a virtual building. Backward movements can also be observed before animations of objects that are potentially dangerous, i.e. to avoid collision with a virtual toy car that is chasing around the floor, or a startled jump back in surprise (Figure 4).

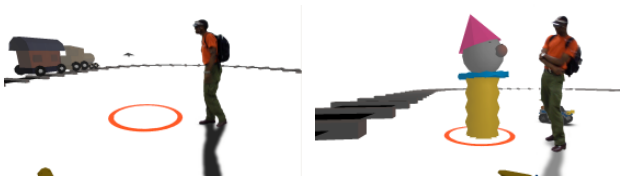


Figure 4. Interaction with the virtual objects.

A shared comment among people observing Empty Museum users is that part of the entertainment is to watch the users' reactions, seeing to what point he or she becomes involved in the virtual world.

Sound plays an important role, given that the spatialization of the different noises generates a sound landscape inside the room, which changes according to the user's position. This way, the user can locate objects of interest through their sounds or produce emotions of anxiety or alarm before threatening noises. In this sense, we must point out the enormous potentiality of the Empty Museum as an audio-based spatial training system.

5. CONCLUSIONS

We have described how we built a wireless, multi-user VR system using "off the shelf" elements. This system solves great part of the constraints found in the conventional VR systems in which the user is connected to a computer through a cable. We have also outlined the characteristics of a new paradigm for three-dimensional navigation through virtual worlds and started to study this paradigm's potentiality.

The system has been tested and assessed by users with and without technical knowledge on VR. From the results of these tests we can conclude that the users' approval has been very good.

The sense of immersion is clearly superior to that reached in preceding systems with reduced mobility. The fact of being able to move and interact as in the real world allows the users to explore the space immediately, without technical skills and without receiving previous instruction on how the system works. Moreover, feedback through the kinaesthetic and vestibular senses introduces the user into the world with a level of immersion that cannot be achieved in any other way. This avoids rejection from the users and makes them very efficient in the use of the system, with a practically null learning curve.

The restrictions of previous paradigms of movement in three-dimensional environments (clumsiness, slowness, impossibility of certain movements, etc.) disappear, offering the user good movement abilities and a capacity of reaction incomparable with any other device or metaphor for simulating movement. The possible size restriction owing to real space dimensions is reduced, if not eliminated, with the possibility of scaling the movement. This allows us to walk around spaces that are much larger than the real physical space available, with acceptably good perceptual results. For larger, more complex spaces, we have also successfully tried out the "teleport chamber" metaphor as a means of transition between different worlds or between areas in one same world.

The described system not only improves the way of exploring conventional 3D worlds, but is also a new way of focussing the design of virtual spaces and VR applications. It opens a new field of research in spatial perception and interaction through the user's own movement, position or viewpoint.

The integration of the Empty Museum with the VRML 2.0 format enables the generation of new multimedia contents. The result of this union of walkability with multimedia is a further advance in VR systems.

6. ACKNOWLEDGEMENTS

Empty Museum was developed by a team of many people that worked closely with the authors during its design, coding and world modelling. Our acknowledgement to them all.

7. REFERENCES

[Bowman98] Bowman D.A, Koller D,Hodges L.F. *A Methodology for the Evaluation of Travel Techniques for Immersive Virtual Environments*. Virtual Reality: Research, Development, and Applications, 1998.

[Cybercompanion] Cybercompanion. http://www.cybercompanion.de.

[Didier01] Didier, S., Dahne, P., Seibert, F., Christou, I.T., Almeida, L., Carlucci, R., Ioannidis, N., *Design and Development Issues for ARCHEOGUIDE: An Augmented Reality based Cultural Heritage On-Site Guide.* International Conference on Augmented, Virtual Environments and Three-Dimensional Imaging, 2001.

[Feiner97] Feiner, S., MacIntyre, B., Höllerer, T. and Webster, T., *A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment.* In: Proc. ISWC '97 (First Int. Symp. on Wearable Computers), October 13-14, 1997, Cambridge, MA. Also as: Personal Technologies, 1(4), 1997.

[Foxlin98] Foxlin, E., Harrington, M., Pfeifer, G. *Constellation: a wide-range wireless motion-tracking systema for augmented reality and virtual set applications*. Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques. p.371-378, July 1998.

[GTK] GTK+ The GIMP Toolkit. <http://www.gtk.org>

[Höllerer99] Höllerer, T., Feiner, S., Terauchi, T., Rashid, G., Hallaway, D., *Exploring MARS: Developing Indoor and Outdoor User Interfaces to a Mobile Augmented Reality System*, In: Computers and Graphics, 23(6), Elsevier Publishers, Dec. 1999, pp. 779-785.

[OpenAL] OpenAL. <http://www.openal.org>

[OpenVRML] OpenVRML <http://www.openvrml.org>

Fast Representation of Implicit Curves Through Space Subdivision

F. MORGADO *and* A.GOMES
IT – Networks and Multimedia Group
Dept. Informatics
Univ. Beira Interior
6201 – 001 Covilhã
PORTUGAL
fmorgado@di.estv.ipv.pt
agomes@di.ubi.pt

Abstract
Current graphical systems include simple primitives to draw line segments, circles, Bézier and NURBS curves and surfaces. But, they have no primitives for general curves and surfaces defined by implicit functions. This paper introduces a fast non-uniform space-subdivision algorithm for implicit curves, without evaluating derivatives and resolving singularities. Intuitively, this algorithm is fast because it tends to only process subspaces containing curve segments.

Keywords
Implicit curves, space subdivision, curve algorithms.

1. INTRODUCTION

An implicit curve C is a level set (or zero set) of some function f from R^n to R, say $C=\{x \in R^n: f(x)=0\}$. In this paper, we present an algorithm to draw implicit curves defined by polynomial functions as in [Blinn82], i.e. algebraic curves, but it may be applied to more general analytic curves. There are three major categories of algorithms to represent implicit curves, namely:

- *Representation conversion*. These algorithms aim to convert an implicit curve into a parametric curve in order to easily display it on a screen [Allgower91] [Bloomenthal88] [Lorensen87].
- *Curve tracking*. It consists of tracking the curve stepwise to find each one of its points, one after another [Moller00][Chandler88].
- *Space subdivision*. Recursively, it splits the ambient space into subspaces, discarding those not intersecting the curve. The subdivision terminates soon after we obtain a good approximation to the curve by a set of small subspaces (e.g. rectangles) [Bloomenthal88] [Taubin93] [Bloomenthal94] [Lopes01] [Triquet01]. Robust approximations can be implemented by using interval arithmetic [Snyder92], algebraic or rational techniques [Krishnan95] [Keyser99], and floating-point arithmetic [Shewchuk97].

This paper deals with a fast graphical representation of implicit curves through curve space subdivision. But, in regions where the curve oscillates or self-intersects, it may be necessary to use a neighbourhood-based curve tracking technique (see [Morgado02] for more details). Anyway, unlike conventional approaches, there is no need to compute derivative and gradient values at every curve point.

2. NON-UNIFORM SPACE-SUBDIVISION

Our algorithm starts by subdividing the 2-dimensional Euclidean space into four quadrants by two intersecting straight-lines, X´ and Y´. These lines form a referential system, called *seed referential*, though they are not necessarily orthogonal (Figure 1(a)). This seed referential must intersect the curve C. Computing the seed referential involves three basic steps:

- Determine its X´-axis by discretely varying the angle $\measuredangle XOX'$, stopping as soon as exactly two points A, C intersect C.
- Set up its origin O´ as the midpoint of the straight-line segment $\overline{AC}$.
- Determine its Y´-axis. Initially, this axis is the straight-line perpendicular to $\overline{AC}$ at O´. But, if it intersects C at three or more points, it is rotated

around O´ until exactly two intersecting points B, D are found (Figure 1(a)).

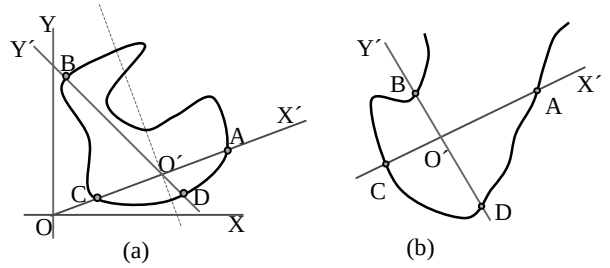


Figure 1: Creation of a seed referential.

The seed referential X´O´Y´ subdivides the ambient space into four sectors, namely: (X´,Y´), (Y´,X´), (X´,X), and (X,X´). Each sector is so defined by its bounding straight-lines clockwise. Note, there may be two or more curve segments in a sector; for example, in Figure 1(b), there are two curve segments in the sector (X´,Y´) provided that the curve is not self-closed.

Found these four preliminary sectors, each sector is recursively subdivided into two subsidiary sectors by some bisector line. Obviously, if a sector does not include any curve segments, it is discarded immediately.

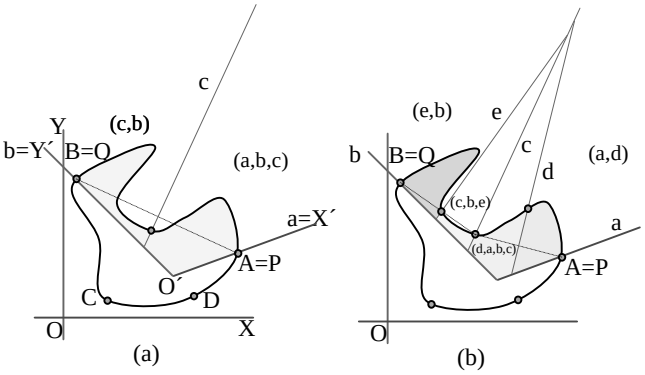


Figure 2:Subdivision of the sector (X´,Y´).

Figure 2 shows how the sector (X´,Y´)=(*a*, *b*) is subdivided into two sub-sectors recursively. The subdivision technique takes the intersection points A=$a \cap C$, B=$b \cap C$ and determines the bisector line *c* perpendicular to $\overline{AB}$ at its midpoint. The result is two smaller quasi-disjoint sectors (*a*,*b*,*c*), (*c*,*b*) (Figure 2(a)). This subdivision recursively proceeds for each subsidiary sector, as illustrated in Figure 2(b). The sector (*a*,*b*,*c*) is subdivided into the sectors (*a*,*d*,*c*), (*a*,*b*,*c*,*d*) by the bisector line *d*, while the sector (*c*,*b*) is split into the sectors (*c*,*b*,*e*), (*c*,*e*,*b*) by the bisector line *e*.

In general, each sector is then defined as a list of bounding straight-lines (r_1, r_2, …, r_n). Its subdivision into two sub-sectors by a straight-line *r* is preceded by the computation of the bounding straight-lines intersecting *r* in the frontier of such a sector.

The number of sector-bounding straight-lines intersecting *r* is two at maximum because any sector is convex. So, subdividing a sector by a straight-line *r* results in two convex sub-sectors. Thus, there are two possible cases:

- *r* intersects a single straight-line r_i bounding the sector (r_1, r_2, …, r_i, …, r_n). The two resulting sub-sectors are (r_1, r_2, …, r_i, *r*) and (*r*, r_i, …, r_n). This is the case of the bisector line *r*=*c* in Figure 2(a).
- *r* intersects two straight-lines r_i, r_j bounding the sector (r_1, r_2, …, r_i, …, r_j, …, r_n). The two resulting sub-sectors are (r_i, …, r_j, *r*) and (r_1, r_2, …, r_i, *r*, r_j, …, r_n). This is the case of the bisector line *r*=*d* in Figure 2(b).

This recursive subdivision for each sector stops soon that both distances *d*(M,I) and *d*(P,Q) are within a small tolerance ε, being M the midpoint point between the sector-bounding points P and Q, and I the intersecting point between *C* and the bisector containing M and I. The value of ε should be as small as possible to get a good approximation to the curve.

Thus, in general, the algorithm is as follows:

ALGORITHM

Begin

(1) Determine the seed referential X´O´Y´
(2) Determine the first four sectors
(3) **foreach** sector (r_1, r_2,…, r_n) intersecting *C* at P, Q
  (a) Determine M ← midpoint of $\overline{PQ}$
  (b) Determine $\vec{v}$ ← vector perpendicular to $\overline{PQ}$
  (c) Determine straight-line *r* defined by M and $\vec{v}$
  (d) Determine list of intersection points $l \leftarrow r \cap C$
  (e) **if** (#*l*=1) and (*d*(M,I)>ε) and (*d*(P,Q)>ε), I∈*l*, recursively call current algorithm for
    (i) sub-sector containing $\overline{QI}$ above *r* in current sector
    (ii) sub-sector containing $\overline{PI}$ below *r* in current sector
  (f) **if** (#*l*≠1) call the neighborhood-based tracking algorithm between P and Q [Morgado02].

End

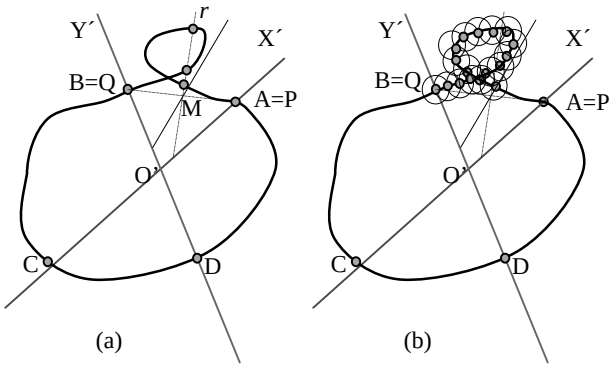


Figure 3: A self-intersecting curve.

3. CURVE SELF-INTERSECTIONS

If the curve *C* self-intersects (Figure 3) or has a local undulation (Figure 4), the bisector line *r* may intersect it at

two or more points. In these cases, a neighbourhood-based tracking algorithm may be called to determine points of *C* between the sector-bounding points P and Q (step 3(f) of the algorithm above).

But, this neighbourhood-based algorithm only starts running if, after rotating *r* around M a number of times by an angle of 10º in current sector, no single intersection point between r and *C* is found. However, if such an intersection point is found, as shown in Figure 3 and Figure 4, the space-subdivision algorithm proceeds for at least one subsector of the current sector.

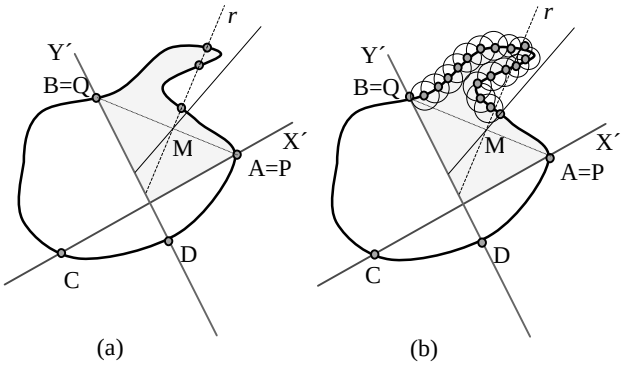


Figure 4: A curve with a local undulation.

4. EXPERIMENTAL RESULTS

Our algorithm for implicit curves was written in C++. Figure 5 shows four curves drawn by our algorithm. The curves (a), (b), and (d) are self-closed, whereas the curve (c) is open. Besides, the curve (d) has a self-intersection. Note that, the curves (a) and (b) were drawn exclusively by the space-subdivision algorithm provided that they are not open nor self-intersecting.

Some experimental results concerning these curves are shown in Table 1. The runtime performance tests were performed on a PC equipped with a 500MHz Intel Pentium, 128MB RAM, and running Windows NT. As expected, the algorithm is faster for curves partially using the neighbourhood-based algorithm as those in Figure 5(c) and (d) than for curves drawn exclusively through the space subdivision technique.

Nevertheless, our space subdivision algorithm seems to be faster than others in the literature. For example, the average number of function evaluations per point for the pure space-subdivision algorithm recently proposed in [Lopes01] is about 23. Our algorithm needs a fewer number of function computations, as shown in Table 1. This is due to the fact that our algorithm makes an irregular subdivision of space. As a consequence, unlike regular space subdivisions, which implies a fewer number of total space subdivisions, therefore a fewer number of points are determined. This also explains why the algorithm performs faster for curves waving slightly than those with constant curvature. For example, a circle takes longer to be drawn than an ellipse because its curvature is invariant, while the curvature of an ellipse slightly changes from a point to another.

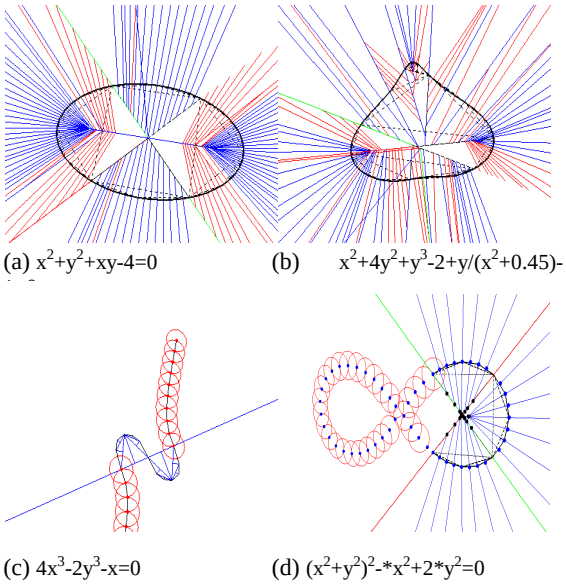


Figure 5: Implicit curves polygonised through our hybrid algorithm.

Figure 5	Function evaluations per point	Drawing time
(a) $x^2+y^2+x*y-4=0$	22.402	0.12s
(b) $x^2+4*y^2+y^3-2+y/(x^2+0.45)-1=0$	19.222	0.11s
(c) $4x^3-2y^3-x=0$	18.231	0.11s
(d) $(x^2+y^2)^2-*x^2+2*y^2=0$	15.210	0.09s

Table 1: Average number of function evaluations per point and overall drawing times for implicit curves.

5. CONCLUSIONS AND FUTURE WORK

A space subdivision algorithm for implicit curves has been proposed. Unlike other space subdivisions, our algorithm carries out a non-uniform space subdivision, which speeds up it somehow. It is also general in the sense it applies to both closed and open analytic curves (not just algebraic curves) with or without singularities. Many multiple-component curves can be also correctly drawn, but that depends on whether at least a bisector line intersects each component.

As future work, we plan to develop further this non-uniform subdivision without using neighbourhood-based add-ons. In principle, this will allow us to extend the space-subdivision algorithm to 3D curves, by using bisector planes to subdivide 3D sectors.

ACKNOWLEDGEMENTS

We would like to thank Professor L.H. Figueiredo for his help with respect to some important details of the algorithm in [Lopes01].

REFERENCES

[Allgower91] Allgower, E. and Gnutzmann, S. "*Simplicial Pivoting for Mesh Generation of Implicitly Defined Surfaces*", Computer Aided Geometric Design, 8(4), pp. 30, 1991.

[Blinn82] Blinn J. F., "*A Generalization of Algebraic Surface Drawing*", ACM Transactions on Graphics, 1(3), pp. 235-256, 1982.

[Bloomenthal88] Bloomenthal, J. "*Poligonisation of implicit surfaces*" Computer Aided Geometric Design, 5, pp.341-355, 1988.

[Bloomenthal94] Bloomenthal, J. "*An Implicit Surface polygonizer*". Graphics Gems, IV, 1994.

[Chandler88] Chandler, R. "*A tracking algorithm for implicitly defined curves*", IEEE Computer Graphics & Applications, 8(2), pp.83-89, 1988.

[Keyser95] Keyser, J., Culver, T., Manocha, D. and Krishnan, S. "*MAPC: a library for efficient and exact manipulation of algebraic points and curves*", In Proceedings of the 15th ACM Symposium on Computational Geometry, pp.360-369, 1999.

[Krishnan95] Krishnan, S. and Manocha, D. "*Numeric-symbolic algorithms for evaluating one-dimensional algebraic sets*", In Proceedings of the ACM Symposium on Symbolic and Algebraic Computation, pp.59-67, 1995.

[Lopes01] Lopes, H., Oliveira, J.B., Figueiredo, L. H., "*Robust Adaptive Polygonal Approximation of Implicit Curves*", In Proceedings of SibGrapi 2001, IEEE Computer Society.

[Lorensen87] Lorensen, W. and Cline, W. "*Marching Cubes: A High Resolution 3D Surface Construction Algorithm*", Computer Graphics, 21(4), pp.163-169, 1987.

[Moller95] Moller, T. and Yagel, R., "*Efficient Rasterization of Implicit Functions*", 1995.

(`http://citeseer.nj.nec.com/357413.html`)

[Morgado02] Morgado, F. and Gomes, A. "*Representing implicit curves with variable resolution.*" Technical Report DI-2002/01, Dept. Informatics, University of Beira Interior, April 15, 2002.
(`http://www.di.ubi.pt/technicalreports.html`)

[Shewchuk97] Shewchuk, J. "*Adaptive precision floating-point arithmetic and fast robust geometric predicates*". Discrete & Computational Geometry, 18(3), pp.305-363, 1997.

[Snyder92] Snyder, J. "*Interval arithmetic for computer graphics*". In Proceedings of ACM Siggraph, p.121-130, 1992.

[Taubin93] Taubin, G. "*An accurate algorithm for rasterizing algebraic curves*". In Proceedings of the 2nd ACM Solid Modeling and Applications, p.221-230, 1993.

[Triquet01] Triquet, F. at al. "*Fast Polygonization of Implicit Surfaces*", WSCG' 2001, pp.283-290, Vol. 2, 2001.

Internet integration of an image mapping system

Jordi Linares Pellicer[§*] jlinares@dsic.upv.es
Jordi Santonja Blanes[*] jsantonja@fractalgraphics.com
Miguel Chover Sellés[+] chover@lsi.uji.es

[§]Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia. EPSA. Plaza Ferrándiz i Carbonell, 03801 Alcoi (Spain)

[*]Fractal Graphics, S.L. Avda. País Valencià, 54, of. 24. 03801 Alcoi (Spain)

[+]Departamento de Lenguajes y Sistemas Informáticos. Universitat Jaume I. 12071. Castellón (Spain)

Abstract

The aim of this work is to present an internet integration of an image mapping system. Applying new textures over real images is a technique with many applications in the industrial field. The purpose of this system is to apply new textures over the different surfaces of objects integrated in an input image, deforming and adjusting the textures properly to fit the surfaces, always considering the original shadows and using digital compositing techniques to get the most realistic result. A general description of this kind of system and a particular solution are presented.

The most important possibilities have been studied in order to integrate this technology into the Internet. We present a solution based on the use of a CGI (Common Gateway Interface).

Keywords

Mapping, warping, Internet, CGI, compositing

1. INTRODUCTION

The image mapping systems are especially useful in some commercial fields and, as almost every computer graphic solution, its integration into the Internet opens new advantages but raises some challenges.

2. AN IMAGE MAPPING SYSTEM

Our image mapping system is based on image deformation techniques, also called warping [Wolberg90]. This deformation must be complemented by considering the original shadows of the input image and mapped over it using compositing methods [Linares96].

The main parts of a general image mapping system are described in figure 6. Our particular solution is presented below.

2.1. Input setting

The input setting is a 2D image where the user needs to apply the textures over some surfaces.

2.2. Masks extraction

The selection of the surfaces from the input image to be covered or mapped is the initial step in our system. These selections can be carried out with manual tools, edge detection techniques or mixed methods [Mortensen95]. The result of this step is a set of one-channel bitmaps (also called masks) with the different areas of the input image to be mapped.

2.3. Warping definition

The most important step of the system is the deformation definition that the textures will suffer to fit the surfaces of the objects[Arad95]. Although there are many image warping methods, most of them are specifically designed for morphing [Gomes99] [Lee95] [Beier92]. In our case, the use of high interactive and predictive tools was essential on the final deformation result. In order to define this deformation, a set of control lines (traced with splines) are used and from which, applying an interpolation method, a quadrilateral mesh is obtained.

The final mesh can fit any kind of surface, just by tracing a proper set of control lines (in 2D). See some examples in figure 1.

2.4. Warping process

The input textures will be deformed following the quadrilateral mesh structure. These textures must be prepared for a tiling disposition and its scale will be adjusted considering setting and texture metrics.

2.5. Shadows extraction and normalization

The extraction of surface shadow maps can be carried out by means of converting the input image to one channel and by simply calculating the intensities [Foley90].

3. INTERNET INTEGRATION

In order to bring the mapping application into the Internet, the first step is to define the requirements the final solution must adopt. Some possibilities have been explored for obtaining the result that best fits those requirements.

Figure 1. Some examples of meshes obtained from control lines

3.1. Requirements

The requirements that an Internet mapping solution must have are [Kolb98]:

- Portability: the web application must run on every browser and operative system.
- Intuitive usage: the first-time user must know how to use the system without further explanation.
- Interactivity: the application must be fast and easy to use. If some data has to be downloaded through the Internet, it must be done in the fastest way minimizing the user's waiting time.
- Quality: the final result must be good enough for satisfying the requirements of the user.

3.2. Possibilities

The technologies to implement such a solution can be classified according to where the main simulation is done: client-side or server-side.

Web application possibilities:

- Client-side
  - Client installed software
  - Downloaded software (Java applet)
    - Local data
    - Downloaded data
- Server-side
  - Image generation (via CGI)

The client-side possibility was discarded. The first time users enters the system they must download the whole program and all the images before beginning to use it, and that process takes too much time. An advantage is that when all data is in the client's computer, the process can be very fast as all calculations are done without accessing the Internet.

A more balanced solution is the server-side approximation, as the user can immediately begin to use the system without downloading a large data set, but the simulation process must be transmitted from server to client via the Internet, but generally it takes a reasonable time for interactivity[1].

3.3. Proposed solution

The proposed solution is a CGI approach, because it has some advantages over other approaches:

- No transmission of a complete program that can slow the process every time the user enters the page.
- No transmission of a huge number of images: if the simulation process is done on the client side, all texture and setting images must be transmitted before the start of any simulation.
- Security of data: only a simulated image will be transmitted.

[1] 10 seconds average with a 56 Kbits connection.

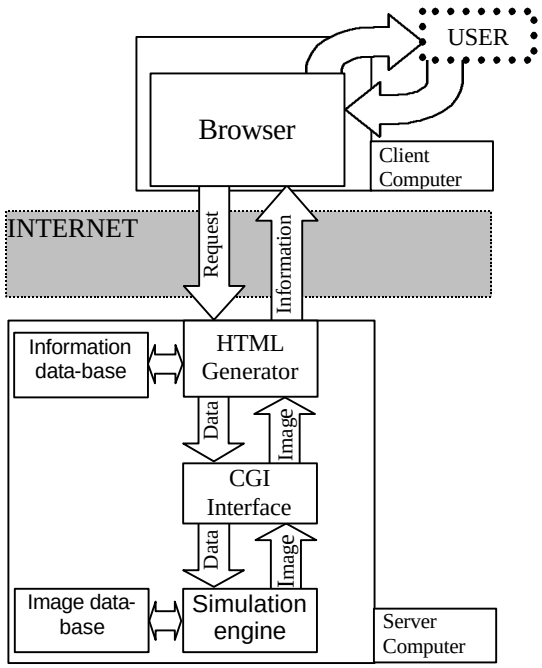


Diagram 1: Internet integration with a CGI

3.3.1. *The general process*

The user enters the main page and selects the texture and setting images to work with.

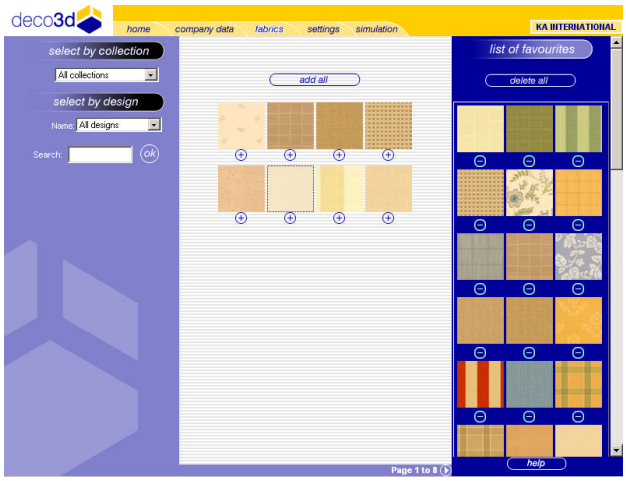


Figure 2. Selecting textures

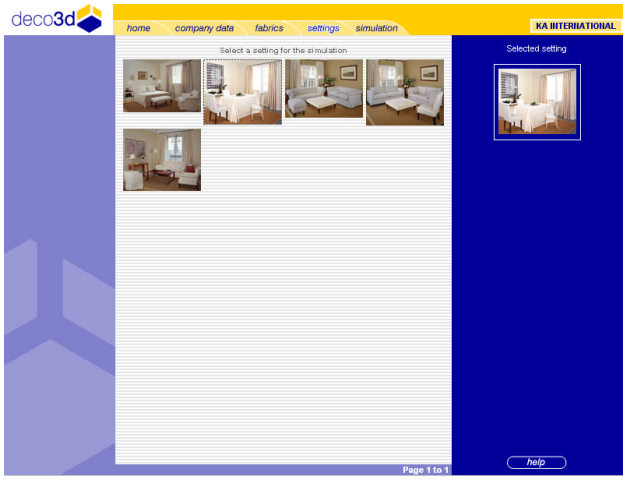


Figure 3. Selecting a setting

After the user has selected the working images, a visual interface must be presented in order to apply textures over setting surfaces.

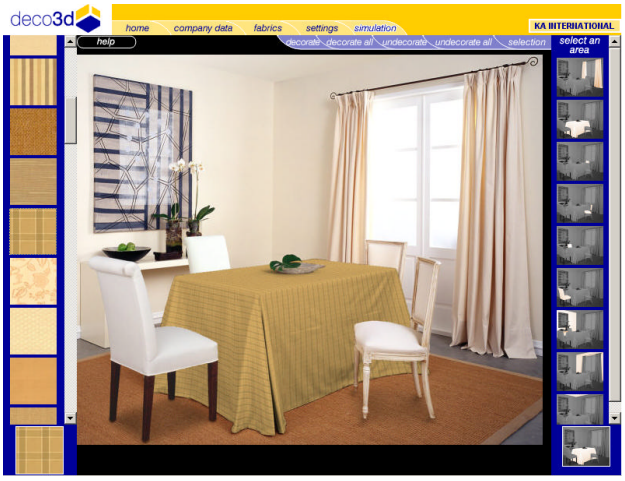


Figure 4. Simulation framework

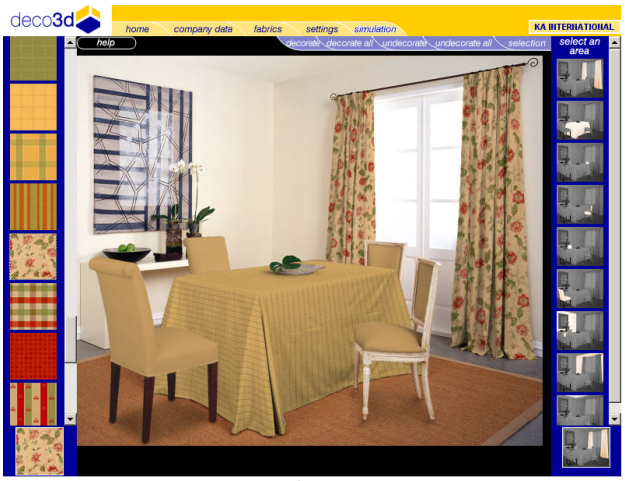


Figure 5. A final simulation

When a selection is ready, the javascript programming issues a request to the server. This request is passed to a CGI in the server that will process it and return an image as a result.

3.3.2. *CGI and the simulation engine*

The CGI decodes the request and performs it. The requested information is usually a setting image on which a set of texture images are applied on some surfaces with a specific distribution. The resulting image must fulfill some requirements such as selected zoom on a specific area, total size and JPEG compression rate.

The CGI calls the simulation core engine with the required parameters. The core loads the images, processes them and generates the resulting image. Naturally, the setting must be prepared before its introduction into the database, with all the masks, warps, normalized shadows, sizes… etc.

Once the final image has been generated by the simulation engine, it is transmitted towards the user's browser. That is a fast process which involves the

transmission of a small quantity of data: the resulting image is normally less than 50-100 Kb.

3.3.3. The core simulation engine

The core simulation engine includes the features of the image mapping system, although the Internet limitations force it to generate high quality simulations at screen resolutions[i].

This is achieved by improving the anti-aliasing method. A medium resolution simulation is internally generated ensuring good results in the mapping process. This simulation is then resampled to low resolution by a fast supersampling algorithm and compressed using JPEG format to avoid going beyond the maximum size file constraint.

4. CONCLUSIONS

The integration of graphic solutions into the Internet raises problems such as overloaded bandwidths, lack of interactivity and time delays.

A image mapping system has been described and its internet integration with a perfect balance between quality and velocity, and a high degree of interactivity adjusted to the current level of web development. The image mapping system integration is carried out using a CGI, a server-side solution, providing the users with highly realistic simulations online and in real-time. The final integration can be tested at <http://deco3d.com>.

5. REFERENCES

[Arad95] Nur Arad, "Image Warp Design Based on Variational Principles", Ph.D. Thesis, Tel-Aviv University, 1995

[Beier92] Beier T., Neely S., "Feature-Based Image Metamorphosis", Computer Graphics, 26, 2, July 1992

[Brinkmann99] Brinkmann, R. "The Art and Science of Digital Compositing", Academic Press, ISBN 0-12-133960-2, 1999

[Foley90] Foley, James D., Andries Van Dam, Steven K. Feiner, and John F. Hughes, "Computer Graphics: Principles and Practice", 2nd Ed., Addison-Wesley, Reading, MA, 1990.

[Gomes99] Gomes J., Darsa L., Costa B., Velho L., "Warping and Morphing of Graphical Objects", Morgan Kaufmann Publishers, Inc. ISBN 1-55860-464-2, 1999

[Kolb98] Andreas Kolb, "Visualization over the Internet", Central European Seminar on Computer Graphics 1998 (CESCG '98), April 1998. <http://www.cg.tuwien.ac.at/studentwork/CESCG98/AKolb/index.html>

[Lee95] Lee S.Y., Kyung-Yong C., Sung Yong Shin, Wolberg G., "Image Metamorphosis Using Snakes and Free-Form Deformations", Proc. Siggraph '95, 1995

[Linares96] Linares, J., "Desarrollo de un Sistema CAD de Simulación 3D de Nuevos Diseños Textiles", Congreso Español de Informática Gráfica CEIG96, Junio 1996.

[Mortensen95] Mortensen E.N., Barrett W.A., "Intelligent Scissors for Image Composition", ACM Computer Graphics Proceedings, Annual Conference Series, 1995

[Smith95] Smith, Alvy Ray "Image Compositing Fundamentals", Microsoft Technical Memo 4, August 15, 1995

[Wolberg90] Wolberg, George, "Digital Image Warping", IEEE Computer Society Press.

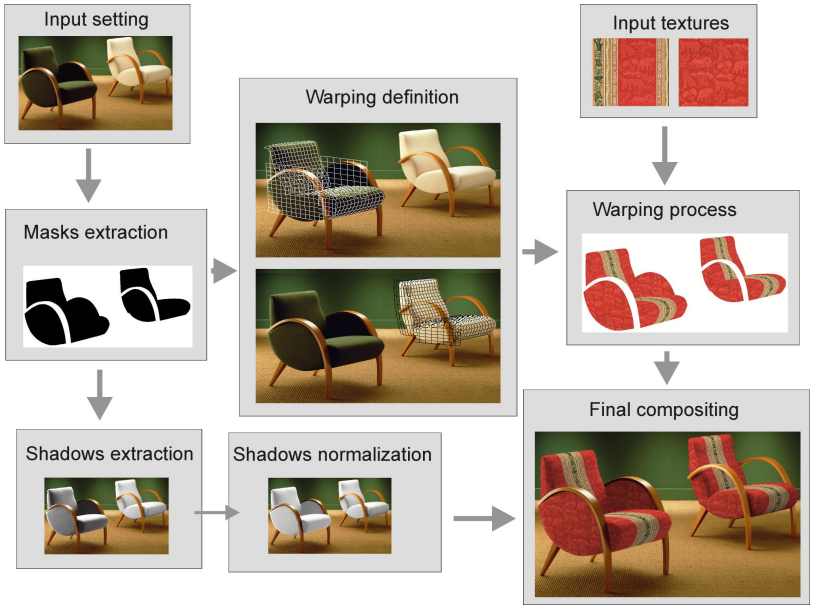


Figure 6. The main parts of an image mapping system

[i] 800 x 600 pixels in <http://deco3d.com>

Presentation Components for Augmented Reality

Adelaide Trabuco
DI/FCT/UNL
Quinta da Torre, 2829-115 Caparica
mart@di.fct.unl.pt

Nuno Correia
DI/FCT/UNL
Quinta da Torre, 2829-115 Caparica
nmc@di.fct.unl.pt

Abstract

When we interact with the natural or urban physical environment it is relevant, in many situations, to have access in real time, to data about what we are seeing. The technologies of Augmented Reality (AR) are based on combining, in real time, synthetic images and real images that provide more knowledge about the real environment around. This paper presents work being done in the ANTS (Augmented Environments) project. The main focus of the paper is the presentation components where the real time video stream is combined with the synthetic elements. These presentation components are part of a more complex architecture for augmented reality that is also described. The paper presents the overall system and the results that were obtained so far.

Keywords

Augmented Reality; Software Architecture; DirectShow; Software Components

1. INTRODUCTION

There are many situations where information about physical locations could be useful but it is not easy to access. The technologies of augmented reality (AR) are based on combining, in real time, synthetic images with real images, giving more information about the real environment around us. A user of AR system sees through a Head Mounted Display (HMD) simultaneously the real world and additional data.

AR lets the user have a better perception of the real world and promotes a more efficient interaction with it. Virtual object composition on real images provides additional insights that the user doesn't detect daily through his senses. This information can help the user to have a better performance on the tasks at hand, minimizing the time to do it, or it can help activities such as the ones related with tourism and learning.

The principal advantages of AR Systems when compared with Virtual Reality (VR) Systems are [Barfield95, Azuma97, Feiner97]:

- The computational resources necessary for composing synthetic images on real images are much less than what is necessary to generate the synthetic scenes.
- Using real images gives a high level of performance and provides a more realistic view of the environment under study or observation.
- The users stay connected with the real world.

There are several complex technical and scientific problems when implementing AR systems, such as:

- The register of synthetic images on real images.
- Position identification.
- Information retrieval.
- Presentation.

The work reported in this paper is a contribution towards the solution of these problems, using, as case study, applications for environmental management. This is being done in the scope of a project named ANTS (Augmented Environments). The main objectives of ANTS are:

- Establish a more appropriate configuration in AR Systems for natural and urban environments. This includes the definition of the overall architecture, considering different devices for access and presentation and the implementation of each of the modules (described in detail in the next section).
- Study new methods for image registry using control points. These methods are based on the definition of geo-referenced control points, identifying key objects in space areas that are being studied.
- Study a AR infra-structure that makes it easier to:
    - Visualize the evolution process of natural and urban environments.

- Provide information about additional elements in the environment.

These objectives are going to be tested in different applications, related with "Parque das Nações" and "Estuário do Tejo":

- Visualization of the water quality of the artificial lakes and the river.
- Information about the buildings and the events that take place on them.
- The composition of synthetic data on images taken from the ground revealing layers of information about the underground below the place where the user is located

This paper is mainly about the presentation components where the final image is composed and displayed, but the overall architecture of the system is also described. Next section is a brief summary of the ANTS project as a whole. The third section describes the application that is being developed. The paper finishes with some preliminary conclusions and directions for future work.

2. ANTS (AUGMENTED ENVIRONMENTS)

As mentioned above, ANTS is an augmented reality system that allows having access to additional information about buildings and other structures in the real world. The initial hardware infrastructure includes a laptop, head mounted display and tracker, a video camera and a GPS receiver and it uses mobile phones for providing the communications. The computational system supports functionalities ranging from image capture and user positioning to data retrieval from remote databases. These functionalities are handled by an architecture that integrates several modules split between client and server (Figure 1). The user will see the results of composing the video image obtained using the camera with the synthetic content retrieved from the server. Extensions to other devices such as PDAs and 3G mobile phones are also planned.

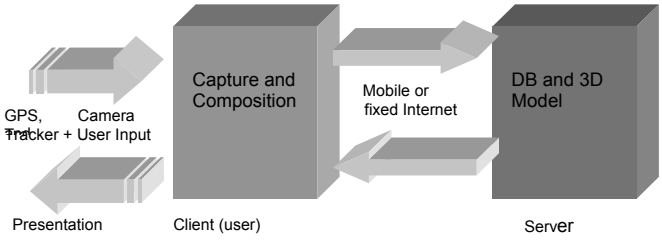


Figure 1. Architecture for augmented reality

2.1 System Architecture

This section describes the architecture of the server side of the ANTS system. The process of enhancing or augmenting the real world video feed is driven by the actions of the user. When the system starts it identifies the position, by combining GPS, tracking and visual cues. After, whenever the user requires information, the systems attempts to locate its position as precisely as possible, and returns the required data. In order to decide which information must be returned the systems stores geo-referenced information in a database and a 3D model of the environment. The materials that are retrieved from the database are then composed into the real image. The main modules that perform theses tasks are the following:

- **3D Module:** The module encapsulates a tridimensional representation of the physical space. It allows to relate the user experience with its computational representation. Using this module it is possible to track and locate the user in both the physical and the virtual world. We have also built a simple tool for editing and adapting 3D models (Figure 2) of the environment that will be augmented.
- **Geo-referenced database:** Keeps the information that will be used to enhance or customize the real world view. Works in combination with the 3D Model in order to locate an element and get the data that is associated with it. The data can be text or graphics, images and even other videos.
- **Image processing/object recognition:** These techniques are used to improve the tracking of the users. Traditional techniques such as GPS are not precise enough and, as such, image recognition of some reference points is used to help the tracking process.
- **Composition and Presentation:** The composition and presentation module combines the image in real time with the data that is retrieved from the database. The 3D module is again used to find the best projection for displaying the information on the video image. The composition and presentation module is described in more detail, in the next sections.

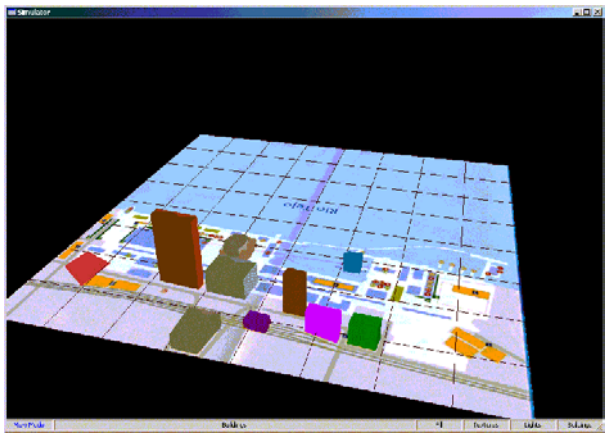


Figure 2. Model Editor

2.2 Hardware Infrastructure

As written above we are using a head mounted display, camera and a tracker. The specific details are the following:

- Creative Labs Video Blaster ® WebCam Go: used to capture the image visualized by the user (in the prototype phase);
- Daeyang E&C Cy-Visor Mobile Personal Display DH-4400VP: used to show the composed image to the user;
- Intersense Intertrax 2: used to capture the movements of the head and adjust the view accordingly.

In the prototype phase we are using a laptop and GPRS phones for communications, but this set up will evolve to use wearable computers [Mann97].

3. PRESENTATION COMPONENTS FOR AR

The main task of this module or application is to combine the image captured in real time with the data that is retrieved from the database. Different media are considered and can be overlaid in the real image: video, image and text, in different formats. The presentation module is a DirectX (more specifically DirectShow) application that uses the filter model as the basis for the architecture. In this model each component has input and output pins and a processing element. If the input and output pins have compatible formats the filters can be connected, defining a filter graph that represents the sequence of operations applied to the input media. We have used pre-defined components and also custom filters, mainly to implement the remote access interface to the server and the image composition.

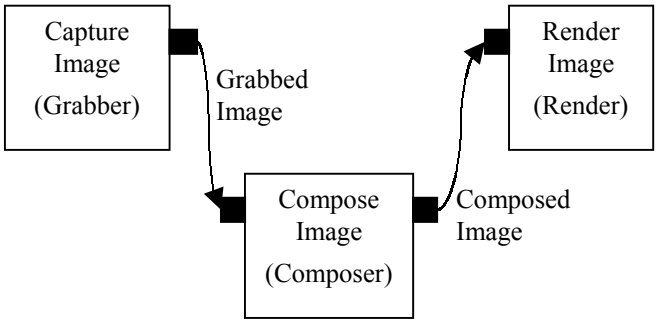


Figure 3. Presentation Module

The architecture used in this module is based on three main components:

1. Grabber: Captures the image in real time from the video camera and delivers it to the composition module.
2. Composer: Combines the image grabbed on the first component with additional information (text, images, audio, and video) retrieved from the server
3. Renderer: Renders the composed image to the HMD glasses.

The first and the third components are standard components provided with Direct Show. The component for composition is a fairly complex one that handles data retrieval and composition and it was developed in the project.

3.1 Image Composition and Presentation

The main tasks and requirements associated with this component are: (1) to read the information about a specific location as an XML file, described below; (2) to combine each of the grabbed images with the data that is described in the XML file.

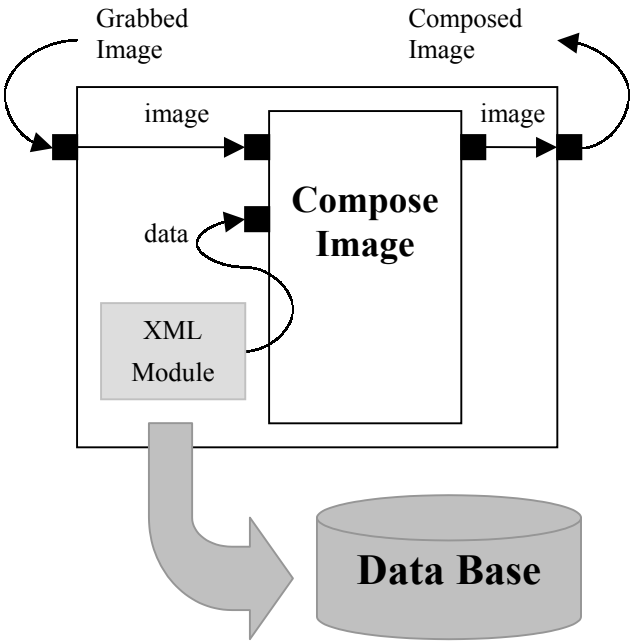


Figure 4. Image composition component

Currently, the external information is stored in a database and in the server file system. The information about each of the objects is available as a URL (Uniform Resource Locator), though HTTP. An example of a query is the following:

```
http://lucas.di.fct.unl.pt/ANTSGod/query.jsp?
SelectObj=10
```

The answer is returned as a XML file, with a predefined format. In order to read it, we have used the tools provided by the Apache Group. Upon reading each of the URLs from the XML file, the data is retrieved from the server database. The simplified example illustrates this type of file. This example considers only one object, with one media type associated to it.

```
<objects>
  <object>
    <id>1</id>
    <name>building1</name>
    <coord x="100" y="100" z="0" />
    <objectmediatype="image" subtype="" >
      <title>overlay-desc</title>
      <format>bmp</format>
      <url>description.bmp</url>
    </objectmedia>
  </object>
</objects>
```

After reading the file, we begin to compose the image with additional materials in the form of video, image or text. Each of these enhancements to the original image is composed in the bitmap retrieved from the video camera. The resulting image is presented fully finished to the user, in order to avoid flickering. The following figure shows an example of one of these images that resulted of the composition of the captured video frame with a bitmap and title images.

Figure 5. Application example

4. CONCLUSIONS AND FUTURE WORK

The ANTS (Augmented Environments) project has the goal of developing a technological AR infrastructure for environmental management. The described system creates new tools for AR that make easier the perception and interaction with the physical space and its own natural and artificial components.

The system architecture is based on the client-server model and it considers a set of independent structural modules with inter-dependent functionalities. The architecture was designed to be flexible and, depending on the capabilities of the available processing power, to move more or less of the functionality to the client side. Current implementation includes prototypes for: (1) edition of 3D models; (2) visualization of synthetic images on real time video; (3) object identification and tracking, (4) server and database. Future work includes the improvement of the mentioned prototypes and the development of additional components, mainly the identification and tracking modules. Additional developments integrating mobile devices and augmented reality [Greenhalgh01] are also planned. In this area, we are considering new forms of expression and media [Correia02], such as shared and distributed storytelling environments that take full advantage of the mobile nature of the devices.

5. ACKNOWLEDGEMENTS

We would like to thank to the other partners and colleagues from GASA/FCT/UNL and DI/FCT/UNL that are involved in the ANTS project (Eduardo Dias, António Câmara, Teresa Romão, Edmundo Nobre, Carlos Santos, Rossana Santos, José Danado, and Luís Romero). This project is funded by FCT/MCT, Portugal.

6. REFERENCES

[Azuma97] R. Azuma. A Survey of Augmented Reality. *Presence*, vol. 6, no. 4, pp. 355-285, 1997

[Barfield95] W. Barfield, C. Rosenberg, Augmented Reality Displays, *Virtual Environments and Interface Design*, New York, Oxford University Press, pp. 473-513, 1995

[Mann97] S. Mann, An Historical Account of the 'WearComp' and 'WearCam' inventions developed for applications in 'Personal Imaging', *Proceedings of the IEEE First ISWC*, Cambridge, Massachusetts, EUA, pp. 66-73, 1997

[Correia02] N. Correia, Personalized Multimedia Information Spaces, *ICIG 2002 (International Conference on Image and Graphics)*, Keynote Talk, HeFei, China, 2002

[Feiner97] S. Feiner, B. Macintyre, T. Hollerer, A. Webster, A Touring Machine: Prototyping 3D Mobile Augmented Reality for Exploring the Urban Environment, *Proceedings of the IEEE First ISWC*, Cambridge, Massachusetts, EUA, pp. 74-81, 1997

[Greenhalgh01] C. Greenhalgh et al, Augmenting Reality Through the Coordinated Use of Diverse Interfaces, http://www.equator.ac.uk/projects/citywide/documents/citywide-paper.pdf, *Technical Report*, University of Nottingham, 2001

Progressive transmission of solid models using a space decomposition hierarchical scheme

Pedro Cano Juan Carlos Torres Francisco de Asís Conde
Dpto. Lenguajes y Sistemas Informáticos
E.T.S. Ingeniería Informática - Universidad de Granada
C/ Periodista Daniel Saucedo Aranda s/n
18071 -. GRANADA (Spain)
{pcano, jctorres, fconde}@ugr.es

Abstract

In the last years several solid model compression methods for multiresolution applications have been presented, most of them using 3D meshes. Octrees are a natural multiresolution representation scheme, although it is approximate. Extensions of classic Octrees that represent polyhedral object exactly have been proposed. These extensions include new terminal node types that contain boundary information of the solid. In this work we present the use of a new solids representation scheme for progressive transmission of the models. The proposed scheme is an extension of classical Octrees that incorporate boundary information of the represented object in the internal nodes of the octal tree. This new scheme can represent polyhedral objects exactly with a smaller storage requirement, and can accelerate basic operations with the model.

Keywords

Solid modelling, Hierarchical modelling, Octree, Multiresolution, Visualization.

1. INTRODUCTION

One of the consequences of the increase in processing and visualization capacity of the present systems is the increase of the complexity of the geometric models that we use. In addition, the development of the distributed systems causes that it is necessary to transfer those models through networks, reason why becomes necessary to increase the speed of transmission and to reduce the storage cost.

Multiresolution models based on meshes of triangles [Garla99][Taubi99] can solve the problem of model transmission, building different levels of detail and transferring in each case the wished level.

Some of the schemes used to represent solids and volumes are based on the decomposition of the space, and use hierarchical structures to store the model.

An Octree is the representation of a model by means of an octal tree structure obtained by recursive divisions of the bounding box of the volume to codify [Meagh82] [Fujim84]. The representation with Octrees allows to perform boolean operations and calculation of properties in a simple way and is a natural multiresolution model, but it is an approximate representation.

Binary Space Partition trees (BSP) divide recursively the space using a plane in two separated half-spaces. Initially created to improve the hidden parts removal process [Fuchs80], they have also been used to represent polyhedral objects exactly [Thiba87]. This scheme offers an unambiguous, but not unique representation.

In this work an extension of the classical Octrees is proposed by means of the inclusion of information of the boundary of the solid not only in the terminal nodes, but also in the internal nodes of the tree. In this way, we are able to use it in progressive transmission of the represented solid.

2. THE MODEL: *SP-OCTREES*

To improve the classical Octrees, hierarchic schemes have been proposed that allow us to obtain an exact representation of polyhedral objects by means of the inclusion of new types of terminal nodes that contain part of the surface of the object, obtaining thus a more compact representation [Brune85][Brune90][Carlb85].

In classical Octrees and the extensions proposed, the internal nodes are those that are not homogeneous with respect to the classification criteria. So, in these nodes the only information appearing is the references to its childrens.

The idea of the proposed scheme, that we have called SP-Octrees (Space Partition Octrees), is based on the inclu-

sion of boundary information in internal nodes that partially defines the object represented in each node of that level [Cano02]. Thus, the information of the boundary faces appears in the upper levels of the tree and it is not necessary to repeat the information in neighbouring nodes that share a face.

When a node is completely in or out of the represented solid we classify it as BLACK or WHITE in the same way as in classical Octrees.

When the intersection of the solid and the voxel is concave, we use a CONVEX node. Formally, a CONVEX node is the intersection of the half-spaces defined by the planes included in it with its bounding box. These nodes allow the exact and univocal representation of a convex polyhedral object.

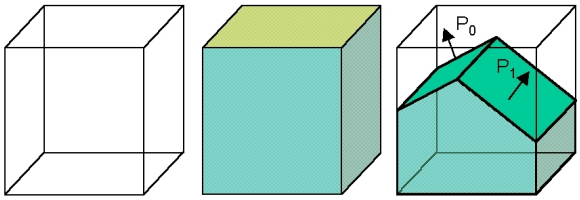


Figure 1. WHITE, BLACK and CONVEX nodes

When the intersection of the voxel and the solid is concave we use a CONCAVE node. Formally, a CONCAVE node is the difference of the bounding box of the node with the intersection of the complement of the half-spaces included in it.

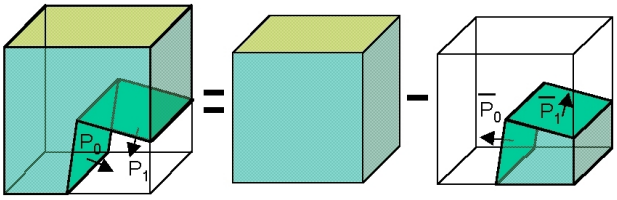


Figure 2. CONCAVE node

When concavities and convexities exist at the same voxel, we classify the node as GREY, dividing it in the same way as in classical Octrees, but maintaining in the node the information of the planes that belong to the convex hull of the solid represented in the node (figure3).

Thus, the solid represented by a GREY node will be the union of the solid represented by each child, but restricted to the intersection of the half-spaces defined by the planes that appear in the father node with the bounding box of this node.

When only one vertex of the represented solid exists in the voxel and concave and convex edges converge in it (see figure 4), we always will have a GREY node although we descend in the tree. Therefore, in order to represent this polyhedron exactly, we needed to include a new type of terminal node.

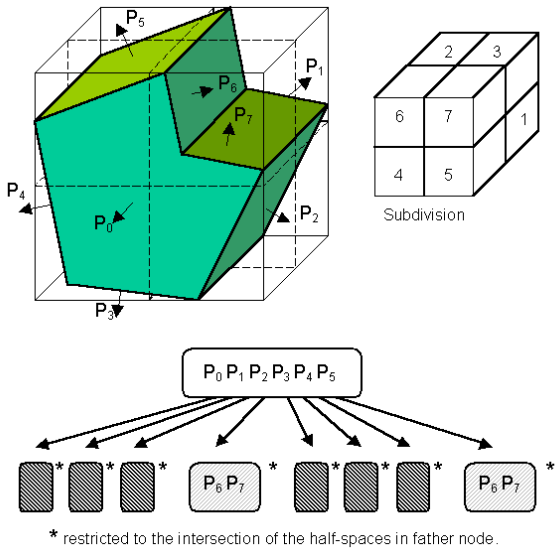


Figure 3. GREY node and its tree.

In these cases we classified the node as VERTEX, storing information on what concave and convex edges appear between the existing planes in the node. So, we will be able to treat it like CONCAVE and CONVEX nodes.

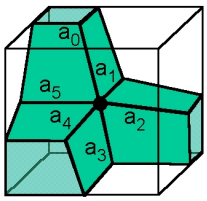


Figure 4. VERTEX node

Any node includes planes that are included in any of its ascendant nodes. The BLACK nodes can be treated exactly like the CONVEX nodes where there are no planes of the boundary of the solid. The WHITE nodes have an empty set of planes.

The CONCAVE nodes will store the reference to the planes that form concavities in the solid. For VERTEX nodes, we treat separately the planes that form concave edges from those that form convex ones. The process itself is similar to the one followed for CONVEX and CONCAVE nodes.

An important aspect of the proposed scheme is that the geometry of the represented object is not stored explicitly, which enables the representation obtained to be compact and reduces storage requirements.

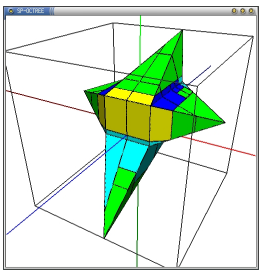
3. BUILDING THE MODEL FROM B-REP

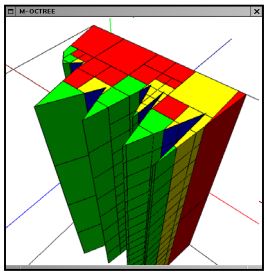
The construction of the proposed structure from a boundary representation of a polyhedral solid, is made by the following recursive algorithm:

1. Compute the bounding box of the solid, which will be the one of the root node of our representation.
2. Classify the node according to planes included in it.
3. If the node is WHITE, CONVEX or CONCAVE, a terminal node with the information of those planes is created.
4. If the node is a VERTEX, a new terminal node is created. The information stored in it is the planes that define each one of the concave and convex edges that share the vertex. So, we can easily recover the geometric configuration of the vertex in the node.
5. If the node is GREY, that is, if concavities and convexities at the boundary of the solid that it represents exist simultaneously, an internal node is created. This node includes the planes that are in the convex hull of this part of the solid, that are not included in any ascendants of this node. Then, we divide the node in eight equal octants, processing each one of them with the remaining planes that form the concavities in a recursive way (step 2).

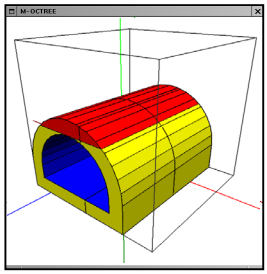
This algorithm is repeated until all the nodes are classified as terminal.

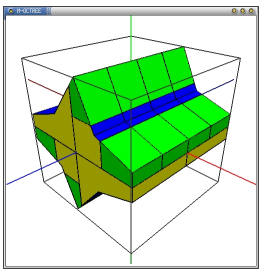
Image 1, 2, 3 and 4 show examples of SP-Octrees of level 2, 3, 5 and 6 respectively, where the blue planes (darker) are those which pertain to planes in concave nodes.





Images 1 and 2. Examples of SP-Octrees of level 2 and 3





Images 3 and 4. Examples of SP-Octrees of level 5 and 6

4. PROGRESSIVE TRANSMISSION

Another of the advantages of the classical Octrees is the inherent arrangement in the scheme, which facilitates the visualization process defining the order of visualization of the nodes. In our case we continue maintaining that arrangement.

In order to visualise an object represented by means of the proposed scheme, we traverse the tree level by level, representing for each node the intersection of the planes that appear in it with its surrounding box and with the planes that appear in their ancestors.

In this way, as we have information of the boundary of the object in the upper nodes, the higher levels of the tree allow us to obtain quickly the convex part of the boundary of the object. To draw the object faces it is necessary to trim the planes in one node against those in its descendants.

As we maintain boundary information of the solid in the internal nodes of the tree (that it is part of the convex-hull of the represented solid in each node), we have an approximated representation of the modelled solid in each level of the tree.

The quality of the approximation improves as we are descending in the tree until arriving at the leaf nodes, where we have the exact representation of the solid.

This mechanism allows us to make a progressive transmission of the tree level by level, so that the receiver of the model can visualize it and operate with it from the beginning of the transmission, without having to receive the complete model.

For each level, we transmit the nodes of the octree that represents the model and the information of the boundary planes that appear in each node of that level (the equations of the planes).

In image 5 we can see an example of the visualization obtained for each level for a concrete solid. The first image of the serie is the original solid (B-rep), and from the left to the right and from top to down we have the representation of each level.

In this image we have the representation of a bunny head (in low resolution by simplicity) in which we can see how the zones of the boundary that need a greater division in the model are those in which a greater concentration of concave edges exists.

We must notice that at no moment we need to transmit geometric information of the polygons that form that boundary, but only the equations of the planes.

As we can see in the example, in the internal levels of the tree, where it only appears part of the information of the solid boundary, we have represented the solid with different levels of detail. This allows us to accelerate operations on the model (for example point's classification or intersection test with a ray).

5. CONCLUSIONS AND FUTURE WORK

In this work a new solid representation scheme has been presented based on an extension of the concept of classical Octree, introducing part of the boundary information

of the represented object, both in the terminal and in the internal nodes.

The proposed method allows an exact representation of polyhedral objects and can be used in progressive transmission of the model.

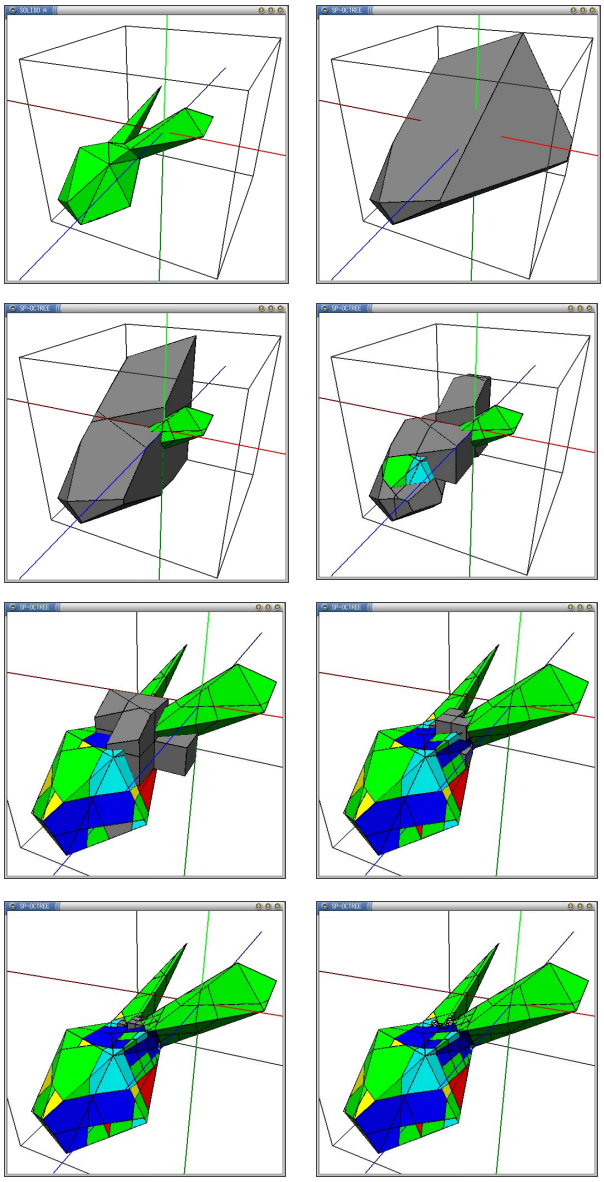


Image 5. Bunny head: original B-Rep model and different resolution levels obtained with SP-Octree

We can store the information of the planes that define the boundary of the solid in an auxiliary structure, and we reduce the repetition of planes in neighbouring nodes that share them (inserting them in inner nodes of the structure). So, the required memory space is smaller than with other schemes.

In addition, we continue maintaining the properties of arrangement of the classical Octrees, and, due to the own orientation of the planes inserted in each node, it is easy to carry out the interrogation and visualization of the model.

We are making a detailed comparative study with other representation schemes, both in space and computation time and operations complexity.

Also, we are studying the compression rate and making a comparison to other schemes for progressive transmission including time for encoding and decoding.

6. ACKNOWLEDGEMENTS

This work has been supported by the *"Ministerio de Ciencia y Tecnología" (Spain)* and *by FEDER* under contract TIC2001-2099-C03-02.

7. REFERENCES

[Brune85] Brunet, P.; Navazo, I.: *Geometric modelling using exact octree representation of polyhedral objects.* Eurographics'85, (1985).

[Brune90] Brunet, P.; Navazo, I.: *Solid Representation and Operation Using Extended Octrees.* ACM Transactions on Graphics, Vol. 9, nº 2, pp: 170-197, (1990).

[Cano02] Cano, P.; Torres, J.C.: *Representation of Polyhedral Objects using SP-Octrees*. Journal of WSCG, vol. 10 (1) pp: 95-101. (2002).

[Carlb85] Carlbom, I.; Chakravarty,I.; Vanderschel, D.A.: *A hierarchical data structure for representing the spatial decomposition of 3D objects.* IEEE Computer Graphics & Applications 5,4, pp:24-31 (1985).

[Fuchs80] Fuchs, H.; Kedem, Z.; Naylor, B.: *On Visible Surface Generation by a Priori Tree Structures.* ACM Computer Graphics, 14(3), pp: 124-133, (1980).

[Fujim84] Fujimura, K.; Yamaguchi, K.; Kunii, T.: *Octree-related data structures and algorithms*. IEEE Computer Graphics and Applications, pp:53-59, (1984).

[Garla99] Garland, M.: *Multiresolution Modelling: Survey & Future Opportunities*. EUROGRAPHICS'99 State of the Art Report. EG, (1999).

[Meagh82] Meagher, D.: *Geometric modelling using octree encoding.* Computer Graphics and Image Processing, 19(2):129-147, (1982).

[Taubi99] Taubin, G.: *3D Geometric Compression and Progressive Transmission.* EUROGRAPHIS'99 State of the Art Report. EG, (1999).

[Thiba87] Thibault, W.C.; Naylor, B.: *Set Operations on Polyhedra Using Binary Space Partitioning Trees.* ACM Computer Graphics, 21(4), pp: 153-162, (1987).

Real time rendering of terrain models: Application to management of aerial resources fighting forest fires

Julián Flores, José Varela, Jose M. Ferro, José A. Taboada, Juan E. Arias
Technological Research Institute.
University of Santiago de Compostela.
15706-Santiago de Compostela. Spain
eljulian@usc.es

Abstract

In this paper, a 3D global information system (3DGIS) applied to fighting forest fires is presented. The application makes it possible to monitor and locate a set of mobile units of a fleet over a 3D terrain model, in real time. The application works in a distributed architecture which links very different hardware and software element, from localization, communication systems to data base and real-time rendering techniques.

Every element of the fleet sends its position, obtained by a global position system (GPS), its state and the information required by a digital mobile phone or trunked radio networks to a communication server. The communication server decodes, processes and sends this information to every monitoring system by a TCP/IP network. The monitoring system or graphic client carries out the several tasks, showing a terrain model of the working area, the updating of the data base with the information of the mobiles and the updates of the terrain model.

The terrain is a three-dimensional model of the topography of the Galicia which can be visualized in real time, with a high level of detail, through the use of L.O.D., polygonal simplification and texture compression techniques adapted to the singularity of PC platforms. The paper will focus on the architecture of the application and implementation and visualization of the terrain model.

Keywords

Location systems, 3DGIS, terrain visualization

1. INTRODUCTION

Since the beginning of the history of the Geographic Information System (GIS) it has been a basic tool to analyze, understand, test and take decisions in different areas. These GIS systems use bi-dimensional cartographic models, onto which the information is shown in some way.

Recently several 3D terrain visualization applications have been developed[Lindstrom96][Koller95]. These applications visualize wide topological extensions with interactive frame rates [Hernandez99]. These applications might be considered 3DGIS, however they present some problems which limit their wide diffusion. They show a limited number of information layers. There are cases in which the application is only a flight over the terrain. They are closed packets which depend on hardware platforms and present communication difficulties with other software. Few systems are able to include information in real time, and only by the pre-processing of information and rebooting the application, new layer can be shown. Finally one important point is that these applications use high end workstations as platforms which means that only governments, military or big companies can pay for.

In this paper, we shall present the first results of a 3DGIS used to locate aircrafts used in the fight against forest fires. The system uses a pre-computed subdivision of data terrain, stored in a data base as a set of blocks of information, formed by a height data mesh, photo textures (aerial or satellite) over it and several layers of information, which are processed separately. The level of detail and the simplification of block mesh, in real time, reduce the number of polygons to be rendered per frame, optimization of the textures includes compression and mipmapping techniques. The platforms used are consumer PC´s.

2. GALICIAN FOREST SERVICE AND AERIAL RESOURCES

The main factor in the deforestation of wide geographical areas is forest fires. Galicia, north-west of Spain, with an extension of 38000 km^2, approximately, with a forest area of more than 13000 km^2 suffers this problem virulently. To fight this plague the forest service focus their efforts one early detection and fast response with adequate resources using geographic information systems and aircrafts. The Galician forest service has several operation centers. Every operation center is linked with the others by an intranet and with the aerial fleet through two mobile communication networks: a public digital mobile phone network and a private trunked radio network. The intranet consists of a set of Data Base SQL servers which control the different operations. The aerial fleet is formed by around 26 airplanes and helicopters,

used to patrol the area and to extinguish the fires. These units are deployed in aerial bases in order to minimize the flight times. The aircrafts are supervised using GPS (global positioning system) working in absolute mode and two, trunked radio and GSM, networks in order to guarantee correct communication and location. The D.B server distributes the information to the different visualization spots.

At the present time, the automatic mobile location application is carried out by a GIS application which uses digital cartography from the Spanish National Geographic Institute and Spot satellite images. The application shows several layers of information in a 2D application

In the next sections we shall describe the new application which replaces the GIS application with a 3D-GIS system in a specific computer which we call Graphic client. This new application has to carry out the same tasks as the old one and increase the visualization quality using a 3D interface.

3. GRAPHIC CLIENTS, VISUALIZATION SYSTEM.

The tasks of the graphics client is to show the terrain model, the mobile units and the information layers that the user needs to supervise the fleet in real time. The architecture of the graphics client is modular and formed by: a data base and data base management system, graphics engine, the network communication and the update module, Fig1.

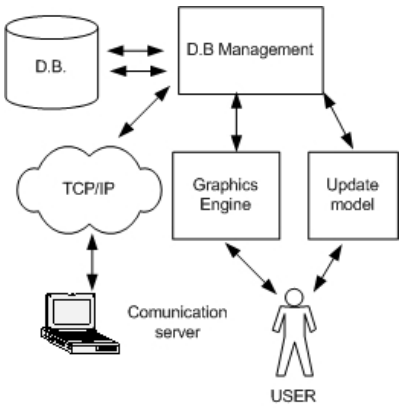


Figure 1. Architecture of the graphics client.

The data base basically stores the terrain model and its different information layers, the mobile unit position and their records. The Data base manager controls the I/O of the DB. The engine carries out the visualization tasks, renders the terrain model and the information over it. The bottle neck of this process is the terrain visualization.

The terrain is typically represented as either height values sampled or a rectangular grid or a network of triangles and a set of photo-textures, aerial or satellite images which give the model a high degree of realism. In either case data sets tend to be quite large and make it difficult to render them in real time because the terrain data can not fit entirely in the memory so they need to be loaded from the disk. Even if the model could entirely fit into the memory, the pipeline does not have enough power to render it in real time. The communication module refreshes data sent by aircrafts: identity, position and state. These modules are presented in the next sections, focused on the design of the terrain model and the graphics engine.

4. TERRAIN MODEL

The data used to build the terrain model have two different sources, one for the polygonal mesh and another for the photo-textures on it.

The source of the polygonal mesh is the “Instituto Geografico Nacional” of Spain. The elevation terrain model of Galicia belongs to the 29 UTM zone and has a resolution of 200m. The information has been processed in order to obtain a regular rectangular grid of 2^n quadrilaterals each one consisting of 2 triangles. The final grid has more than 10^6 vertices. The regular rectangular grid is necessary in order to apply the optimization techniques as we will explain in the next section. This grid is orthogonal and geo-referenced.

The textures layer was created making a composition of orthophotos from Landsat and Spot satellites. The composition takes advantage of the visual quality of Landsat images (true color) and the resolution of Spot images (10m pixel).

4.1 Engine design

The engine must render the model in real time. When we analyzed the model, a significant problem to be taken into account was that it had more information than the PC-platform could render in real time. Therefore several optimization techniques had to be applied to obtain the adequate frame rate. Firstly a pre-rendering technique was applied. The terrain model has been pre-processed by division in blocks. Every block is composed by the height grid and its corresponding texture. Each block is independently processed and only a number of them are rendered in every frame, the invisible area is hidden by a foggy effect. The election of the block size, that blocks are visible, their optimization and the switching of these blocks when the user moves the camera are the bottle neck in the visualization process.

The number of blocks visualized has to be a balance between the time necessary to load of the blocks and the number of blocks switched per frame. The number of blocks switched depends on the size of the block, the speed of movement of the user and the size of the view-frustum. For a maximum speed of 10m per frame with a frame rate around 24 frames per second with a view-frustum of 60º, at distances 30.000m and 100m, a flight height of around 1000m, the hardware and the simplification algorithm use blocks of 12.800*12800m. Another quality of this size is that it produces sub-grids or blocks with a number of 2^n vertices which leads to fast and easy simplification. The information of the vertex shared by two consecutive blocks is repeated in each one in order to provide gaps between consecutive blocks.

In real time, if one terrain block is in the view-frustum, it must be rendered. Nevertheless every visible block does

not have to be rendered with the highest resolution possible because for a block faraway from the position of the camera, two consecutive vertices are represented over the screen on the same pixel. Therefore, the faraway blocks are approximated by a lower resolution, increasing the rendering speed by decreasing the amount of triangles to be rendered. In order to implement the simplification of, polygonal and texture optimization technique has been applied.

4.2 Polygonal optimization

From the point of view of polygonal optimization several algorithms have been developed in the last few years. These algorithms, for example ROAM or Lindstrom, drastically reduce the amount of triangles rendered per frame by the graphic pipeline but for their applications a very powerful CPU, a very wide bus and very fast disks to transfer this amount of information to the graphic system are necessary [Duchaineau97] [Lindstrom96] [Stewart97]. These transferences are usually the bottle neck of the process because the bus is not wide enough or the disk fast enough to I/O the information in real time. These algorithms were not designed to optimize their run in the new hardware graphics in special PC graphics cards where the calculus power of the CPU and graphics pipeline grow every 6 months while the width of the bus is stabilized in several consecutive generations of hardware. In this way, it is necessary to apply more conservative algorithms and new programming techniques which obtain the best performances from the new PC graphics pipelines and their GPUs.

The algorithm used for the polygonal simplification of the grid is geo-mipmaps [Boer00]. This algorithm is an extension of the mipmapping technique for textures to simplify polygonal grids. The process of mipmapping, takes values of one pixel from the combination of adjacent pixels, smoothing the transition from one mipmap level to the next. In this way, a set of vertices could be represented by only one, its value being the combination of the originals These algorithms have been used before, the different levels of geo-mipmaps are pre-calculated and stored. Therefore it is possible to carry out this polygonal simplification in run-time by the utilization of the new extension of the pipelines graphics [openGL]. The simplification is then faster and consumes less memory so only one copy of the model is necessary, decreasing the space in HD, switch-time and the memory used is minimized. The extensions make it possible to manage the models like arrays of data and it is possible to specify which of the elements of the array (vertex) are rendered for face or not by the inclusion of a simple offset. This procedure is very fast and defines LOD from the maximum resolution to 4 vertices per block with only the specification of one parameter (offset), and the L.O.D. can be defined for a 2^n grid in n/2-1 levels. In our case only 4 levels of detail have been taken into account.

The choice of the level of detail of a block marks the good performance of the rendering process. In most of the 3D system the distance is the main parameter. We have included the frame rate as a part of the equation, so the user never has a feeling of frustration waiting for the next frame. The system is more portable, it can work on different platforms but with higher or lower precision.

One problem when the engine works with a model terrain subdivided in blocks is the geometry gaps. When two blocks of terrain have different levels of detail, cracks at the edges of the block will be visualized. The simple way to solve this problem is to add extra vertices at the edges. In this way the vertices of the edge of the block are not simplified form one level of detail to the next, only the interior vertices are simplified. Therefore, Boer[Boer00] proposes another alternative algorithm to solve the geometric gaps by the duplication in the connectivity of the vertices Fig 2. This technique renders two triangle-fans per edge of the block. The remaining vertices are drawn in the normal way. The process must be applied to each of the four edges of the block.

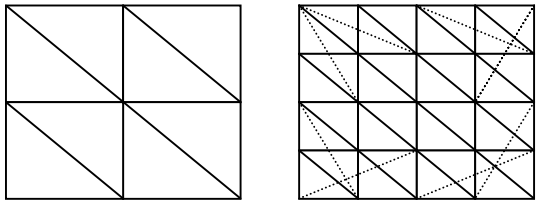


Figure 2 left low detail, right high detail plus new faces.

4.3 Texture optimization

Texture optimization is one of the more significant problems when the engine has to work with large amount of images or they are too big. The most common technique used in model terrain rendering is clipmapping [Tanner98]. The problem of this technique is that it requires hardware support and it does not have software emulation and nowadays the graphics consumer pipes do not support this technique.

In our case, in order to minimize the texture memory used (251Mb), compression texture has been used. The OpenGL texture compression extension offers a tremendous boost for the rendering pipeline at different levels: faster rendering, lower texture memory requirements, faster texture downloads into texture memory, lower disk space requirements for storage and faster disk access. The method used for the texture compression is S3TC which can be used directly with OpenGL[Opengl][Nvidia] and the compression of the textures is developed in a pre-render step. This reduction of the texture size decreases the swap time of the textures and the time of the set up process. For example the total size of the satellite image without compression is of 802Kb, and the texture with compression is much lower 171Kb. The visual difference between compressed and non-compressed is virtually negligible. The final amount of memory used is 51Mb.

5. CONCLUSION.

The system has been used in the summer campaign 2001 of forest fire-fighters in an experimental mode. The monitoring system used was a PC-III, 600 MHz with 768M of memory and a graphics pipe based on a NVIDIA Gforece256 with 32M of memory. The final

resolution on the screen is 1280*1024, and the frame rate remains above 20 frames per second. The system receives information from the mobile units and they are represented over the terrain model in real-time. The user can "travel" thought the model of the terrain freely with a stable frame rate and check the position and state of the fleet better than a map table or the initial GIS system used in the control center Figure 3,4,5.

The graphic engine leads to a rendering with a stable frame rate to visualization in real time. In this way the performance of the system is similar to another application working with a high end workstations.

The compression textures techniques minimize the size of the textures on the disk, the memory transfer time and memory of textures used.

The engine has been tested on another platform, the system has been run in a notebook, AMD 500 with an ATI 4M graphics pipeline and 128M of memory, the system only obtained a stable frame rate with a reduction in the resolution of the textures. Nevertheless the fire-fighter thinks that this resolution is enough for the outdoor tasks.

The engine has been checked with other terrain models, areas with different extension and resolution. The algorithm used has obtained real-time performances.

6. ACKNOWLEDGEMENTS

To the Tele-detection group of the System Lab of the technological research institute for the processing and implementation of the satellite images.

7. REFERENCES

[Boer00] de Boer W.H. Fast terrain rendering using geometrical mipmaping. 2000. www.flipcode.com/tutorials/

[Blow00] J. Blow. Terrain Rendering at high level of Detail. White paper, Bolt Action software. www.bolt-action.com

[Duchaineau97] M. Duchaineau, M. Wolinsky, D. E. Sigeti, M. Miller, C. Aldrich, M. Mineev-Weinstein ROAMing terrain: Real- time optimal adapting meshes. UCRL-JC-127870. www.llnl.gov/graphics/ROAM/

[Hernandez99] L. Hernández, J.Taibo, A. Seoane Una aplicación para la navegación en tiempo real sobre grandes modelos topográficos. CEIG99, 287-302

[Koller95] D. Koller, P.Lindstrom, W. Ribarsky, L. Hodges, N. Faust, and A Turner. Virtual GIS: A real-time geographic information system. Georgia tech. Technical report. 95-14gt.

[Lindstrom96] P.Lindstrom, D. Koller, W. Ribarsky, L. Hodges, N. Faust, and A Turner. Real-Time continuous level of detail rendering of height fields. Proceeding of ACM Siggraph 96, August 1996, pp 109-118.

[Microsoft] www.microsoft.com/directx

[Moller99]Tomas Moller, Eric Haines A.K. Real-time rendering. Published by A K Peters, 07/1999, ISBN 1-56881-101-2

[Nvidia]www.nvidia.com\opengl

[Opengl] www.opengl.org

[Stewart97] A. James Stewart. Hierarchical visibility in terrains *Eurographics Rendering Workshop*, June 1997, pp 217-228.

[Tanner98] Tanner, C. Migfal, C. Jones, M. "The clipmap: A virtual Mipmap" *Computer Graphics (SIGGRAPH '98 Proceedings)*, pages 151-158, July 1998

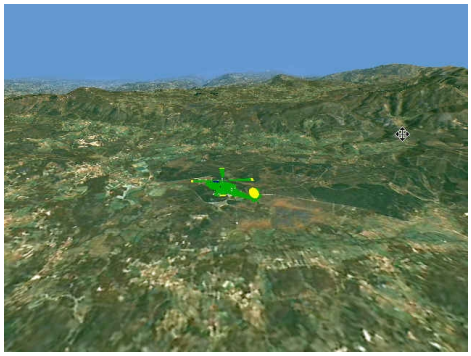


Figure 3. 3D map with one helicopter

Figure 4. 3D satellite map blends with maps roads.

Figure 5. 3D road map.

Simulator for training in Flow Cytometry: Conceptual Model

Filipe Santos
Escola Sup. de Educação de Leiria
Leiria
fsantos@esel.iplei.pt

Beatriz Sousa Santos
Univ. de Aveiro- DET/IEETA
Aveiro
bss@ieeta.pt

Filipe Sansonetty
IPATIMUP
Porto
fsansone@ipatimup.pt

Joaquim Sousa Pinto
Univ. Aveiro- DET/IEETA
Aveiro
jsp@ieeta.pt

Abstract

In this paper we present the conceptual model of a simulator, under development, to simulate a flow cytometer, a very sophisticated and expensive piece of equipment used in biology and medicine. This simulator, along with other tools, is supposed to provide a virtual environment for training in flow cytometry. It offers the trainees and trainers the possibility to simulate not only parts and processes of the flow cytometer itself but also activities in the wider laboratorial environment, such as preparing and studying virtual cell samples or specific training modules.

Keywords

Simulator, Flow Cytometry, training.

1. INTRODUCTION

Today both in Medicine and Biology many different techniques to measure cell characteristics are used. One of these techniques, whose applications are rapidly expanding, is flow cytometry [Shapiro95], which uses sophisticated and expensive instruments, known as flow cytometers, to attain this objective.

Training people in this technique involves several problems:

- Often the number of people having the needed skills to perform this task is not enough;

- The number of flow cytometers in laboratories is usually insufficient due to their high price;

- The few existing institutions that can offer training in this area carry out mainly analysis and investigation and not training (since the benefits are not so evident and immediate);

- Training sessions with a flow cytometer are performed with small groups of trainees, and often are far from the training institution;

- The experts that can act as trainers often face problems of reduced availability due to other commitments.

Trainers also indicate other disadvantages of the present training methods: for example, to offer training sessions to less experienced trainees involving an expensive instrument having high maintenance is, at the best, a calculated risk. Therefore, these facts explain and justify the development of better and more adequate training processes through non-expensive facilitating tools.

Interactive learning and training through simulators is a powerful pedagogical tool that allows trainees to learn, based on experimentation, through the interaction with a carefully controlled, but realistic, environment [McIndoe02]. Several examples of simulators currently being used in medicine for learning and training, among other activities, can be found in [Chung00] [Dubois95] [Burdea99] [Weidenbach00] [Tsai01].

Thus, and aiming at facilitating the teaching, learning and training in flow cytometry we are currently developing a Virtual Environment for the learning and training of flow cytometry based in a flow cytometer simulator, which has a twofold purpose:

1- to provide trainees with a cost effective interactive tool, which can be used as often as they wish, facilitating their learning/training and decreasing, in an initial phase, the need to occupy a real equipment and a trainer

2- to provide trainers with the possibility of preparing simulated experiments, which can be included in multimedia courseware either for presential training or long distance learning/training sessions.

2. VIRTUAL ENVIRONMENT TO USE THE SIMULATOR IN TRAINING

Designing an intuitive, easy to learn and to use tool with such complex and demanding objectives, to be used by trainees with different profiles and needs, and mainly to be used in a distance-learning context, is an interesting challenge.

It is important not to focus only in the development of an accurate flow cytometer simulator, but also to give the trainee, as much as possible, the feeling of being assisted by expert professionals, as cytometrists and trainers, when interacting with the simulator. Hence, it is important to insert in the virtual environment not only the object of study (the flow cytometer), but the actors that would be present in a real training session scenario.

We have proposed a model (fig.1) where trainers and trainees can interact with the training environment and communicate between them. There are two applications involved in this scenario: The trainee will use the *Virtual Environment for the Training of Flow Cytometry* to access not only a virtual flow cytometer, but also other set of tools to help him learn flow citometry. Some of these

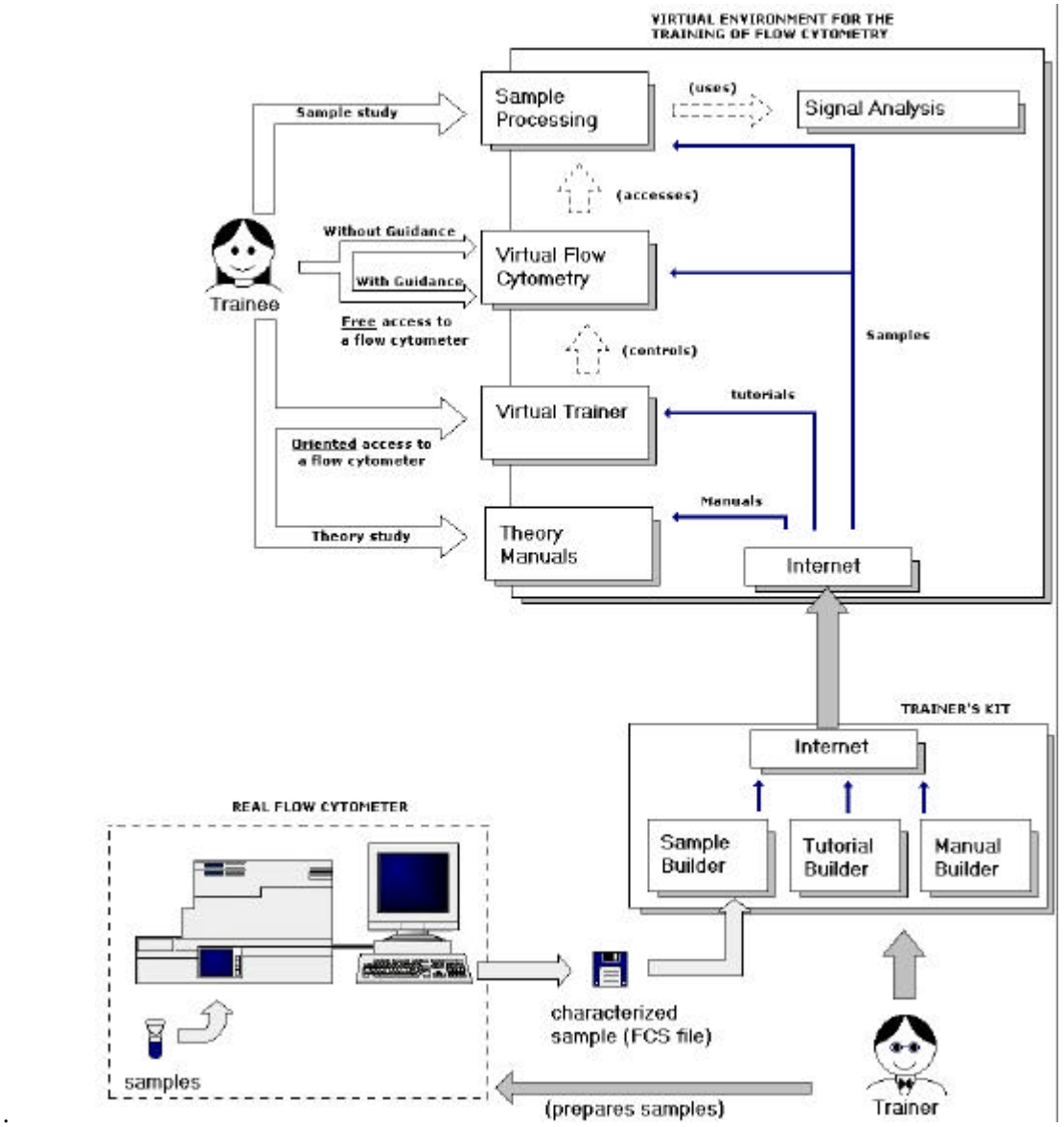


Figure 1: Model of the interaction and communication between trainers and trainees.

tools are aimed specifically at training in using correctly the flow cytometer. Others are more general, and aim at teaching flow cytometry fundamentals and theory. The trainers use another application, the *Trainer's Kit* and have the responsibility of developing several types of pedagogical materials (samples, tutorials and manuals) that the trainee will access to use the simulator to its full benefits. Some of these materials (The *Virtual Samples*) are indeed data collected from real samples in a real flow cytometer. Here the aim of the trainer is to prepare samples that have pedagogical interest in a real flow cytometer and then use the *Trainee's Kit* to insert information, readable by the *Virtual Environment for the Training of Flow Cytometry* that will help the trainee in the task of interpreting the results obtained.

In Section 3 we will explain some other important materials developed by the trainer and their relevance in the training context.

3. THE SIMULATOR

3.1. The Conceptual model

To attain this simulation, the application offers the user a graphic environment that reproduces the essential components of a general flow cytometer, a laboratory workbench and a classroom. The metaphors of the Laboratory Workbench (fig.2) and Classroom (fig.3) were chosen to give the user an interface with high degree of leamability [Mayhew92] [Dix93].

Three modes of accessing the Virtual Flow Cytometer are offered to the trainees (fig.4):

- the first one was meant to give the trainee a free access to the flow cytometer. In this way, the trainee may interact freely as he would with a real flow cytometer and feel the intrinsic difficulties of the instrument and the relations between every component;

- the second mode is a less flexible, more guided way of interacting with the instrument, through the use of tutorials prepared by the trainer, which would guide the unexperienced trainee through the basic aspects and sequences of the flow cyometer operation;

- Finaly, in the third mode, the trainee can study cell samples in the virtual flow cytometer and perform some analysis on them, independently of being capable of operating the flow cytometer; thus, samples can be studied, and flow cytmetry can be learned even if the trainee has never interacted with such an instrument.

As mentioned, the trainer is a part of the scenario by developing pedagogical files that will be used by the trainee. One of these materials is the *tutorial* that acts as a virtual trainer in the trainee's application.

In these tutorials (fig.5) the virtual trainer (a character) guides the trainee through a "lesson" (tutorial) and "controls" every user input in the virtual flow cytometer.

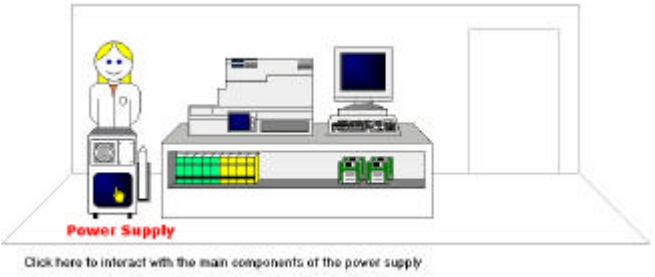


Figure 2: The Laboratory Workbench Metaphor.

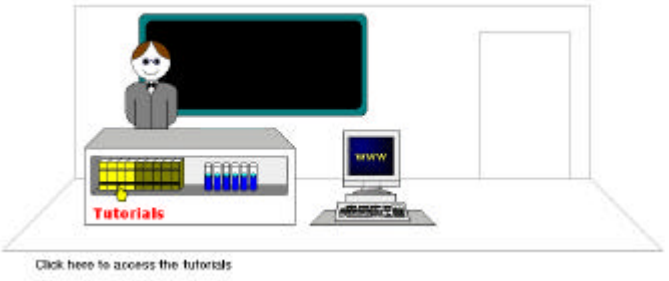


Figure 3: The Classroom metaphor.

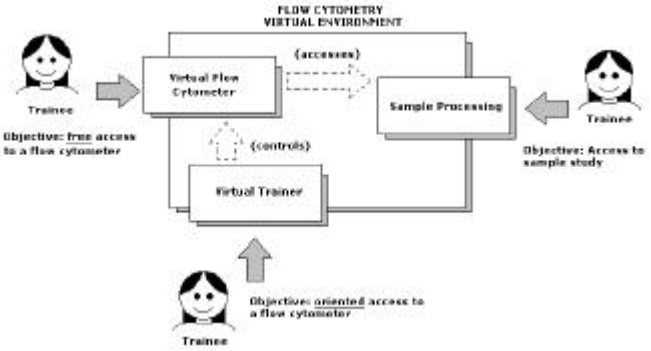


Figure 4: Three access modes offered to the trainees

The virtual trainer instructs the trainee on every operation in a sequence to achieve a certain objective (e.g. switch and calibrate a power supply). Therefore, in this way, a trainee has the possibility of training and learning fundamental aspects of flow cytometry and flow cytometers through distance learning courses [Horton00].

3.2. The Prototype

A prototype was developed for *Windows* platforms using Visual Basic and its ActiveX component characteristics.

This intended an easy migration of the prototype to a Web page if the Virtual Environment proved to be useful and pedagogical. As a consequence the Virtual environment could be accessible by the Internet, and new versions of it would be more rapidly disseminated.

The design and test of the prototype has been actively participated by a final end-user (an experienced cytometry trainer). Informal feedback was obtained from, and simple tests were performed by, other 3 experts and 9 students. Formal usability assessment is still to be obtained through questionnaires and observation techniques applied to a greater number of trainees and trainers.

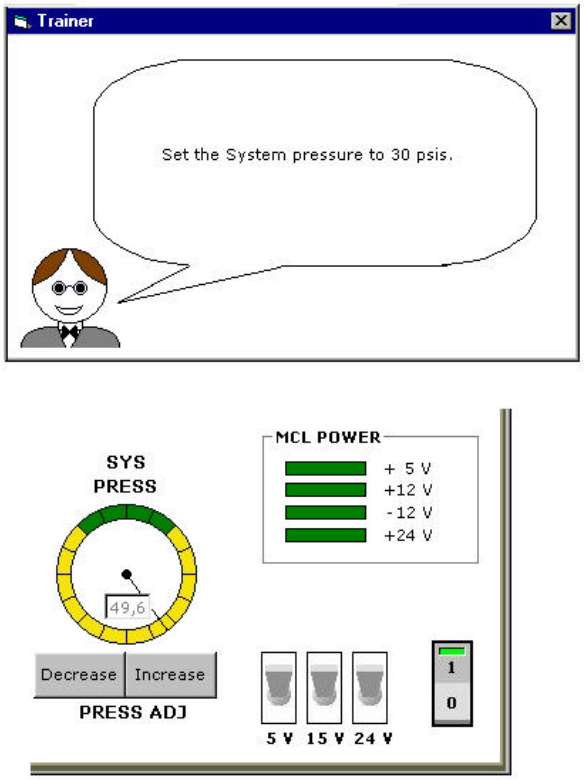


Figure 5: Tutorial Interface for the trainee.

3. CONCLUSION AND FUTURE WORK

In this paper we have presented the main aspects of the conceptual model of a flow cytometer simulator. The training using the simulator is not intended to replace the presential training with a trainer and a flow cytometer, but it is meant to enhance its efficiency and efficacy.

This work was based on an existing prototype [Silva98][Sansonetty98], however the conceptual model has been much enhanced, new functionality has been introduced and the use of the simulator in scenarios of distance learning and training has been devised.

Formal evaluation has yet to be done. The performance of the existing prototype in making easier the leaming and training of future flow cytometrists will be assessed at IPATIMUP with several types of prospective users.

4. REFERENCES

[Burdea99], Burdea, G., G. Patounakis, V. Popescu, R. Weiss. Virtual Reality-Based Training for the Diagnosis of Prostate Cancer. *IEEE Transactions on Biomedical Engineering*, IEEE, 1999, pp. 1253-1260

[Chung00],Chung, A. Development of an Interactive Multimedia Training Simulator for Responding to Abortion Clinic Bomb Threats. *IEEE Trans on Information Techn. in Biomed*., IEEE, 2000, pp. 79-82

[Dix93], A. Dix, J. Finlay, G. Abowd, R. Beale. *Human-Computer Interaction*, Prentice Hall

[Dubois95], Dubois, P., J. Rouland, C. Chaillou, S. Karpf, P. Meseure, F. Duquenoy. A training simulator for retinal laser photocoagulation. *Proceedings of IEEE Engineering In Medicine and Biology*, 17th Conference EMB95, 1995, pp. 779-780

[Horton00], W Horton, *Designing Web-based training*, John Wiley & Sons.

[Mayhew92], D. *Mayhew. Principles and Guidelines in Software User Interface Design*, Prentice Hall.

[McIndoe02], A. McIndoe. *The Future Face of Medical Training*, January 2002 <http://www.multimed.co.uk/media/mcindoe.pdf>

[Sansonetty98], F. Sansonetty, Robinson, P, Lau N, Sousa Santos B, Alves R, Silva L, Sousa Pinto J. Simulations for Facilitating Self-Training in Flow Cytometry: Discussion of the Concept and its Utility. *Proceedings of the Euroconference98: New Techn. in Higher Education*, Univ. de Aveiro, 16-19

[Shapiro95], H.M. Shapiro. *Practical Flow Cytometry*, 3rd Ed. Wiley-Liss, New York

[Silva,98], Silva, L., R. Alves, B. Sousa Santos, N. Lau, F. Sansonetty, J. Sousa Pinto. Simulator of a Flow Cytometer: a Prototype. *Revista do Departamento de Electrónica e Telecomunicações*, Universidade de Aveiro,Vol. 2, nº3, Setembro 1998, pp. 321-324

[Tsai01], Tsai, M., M. Hsieh, S. Jou. Virtual Reality orthopedic surgery simulator. *Computers in Byology and Medicine*, Pergamon, 2001, pp. 333-351

[Weidenbach00], Weidenbach, M., C. Wick, S. Pieper, K. Quast, T. Fox, G. Grunst, D. Redel. Augmented reality simulator for training in two-dimentional echocardiography. *Computers and Biomedical Research*, Academic Press, 2000, pp. 11-22

Stereoscopic views of 2D fractal sets

Rafael Mullor
Data Center, UPV
Univ. Politécnica de Valencia
rmullor@cc.upv.es

Roberto Vivó
CGS/DSIC
Univ. Politécnica de Valencia
rvivo@dsic.upv.es

Abstract

In this paper we show how it is possible to create stereoscopic views of two-dimensional fractal set. Our method simultaneously calculates the two views of the stereo-pair, making a horizontal displacement of the computed value according to its speed of escape. The method is suitable to any set as long as it covers the plane with regions defined by consecutive integer numbers.

Keywords

Fractals, stereoscopy, anaglyphs

1. INTRODUCTION

More than fifteen years have passed since Mandelbrot decided to represent graphically fractal sets of the complex plane assigning colours to each point of raster [Peitgen86]. Since then, many works have appeared using those fractal images to create extraterrestrial landscapes or like a form of computer art. Nowadays, there are many programs in the net for the computation of the neighbourhood of Julia set and boundary exploration of the Mandelbrot set. Those pictures are generated by means of the iteration of a complex function that diverges outside the set. The divergence speed is measured and assigned to the generating point of the iteration. According to the number of iterations a different colour is chosen. The result is a set of regions of homogenous colour. The neighbour of a region is another one with a consecutive colour index. These regions become narrow and twisted more and more as they approach to the attractor set. Therefore, a faster variation of the index is expected near the set.

The stereoscopic views are very old in photography [González02]. The generation of such views in computer graphics supposes the variation of the point of view in three-dimensional scenes. The point of view is located in each one of those positions that would occupy the eyes of the observer. In this way, we have two slightly different images one for right eye and the other for the left one. How the images are provided to the eyes depends on the characteristics of the layer. For example, if it is in paper, the anaglyph technique is commonly used. Other possibilities using the computer are the use of page flipping or interleaved image. In any case, appropriate glasses are necessary to allow each eye to see a different image. With this, a sensation of much more real depth is obtained than using the traditional three-dimensional projection.

2. METHOD

In this work we have used several stereoscopic techniques for the exploration of the sets of Mandelbrot and Julia although the method is extensible to any set of regions where a consecutive index can be assigned to them. The method has served as an artistic way to observe the mathematics as well as a benchmark for the stereoscopy library [XplorA02] developed at our computer graphics laboratory.

As it is known, the Mandelbrot gallery of images is generated by means of the iteration of the function $Z_{n+1}= Z^2_n+C$ where C belongs to the complex plane. C is the generator point of each iteration. To practical effects of representation, it is made a correspondence between C and the pixel by means of an isomorphism between the complex plane and the raster. The colour of C settles down depending on the necessary number of iterations so that the module of Z surpasses an established value. Choosing suitably the colours and the interval on the complex plane it is possible to obtain artistic images. Another possibility is to use the indices as a height-map to generate fractal landscapes.

Nevertheless, it is possible to produce, with the same effort, an image with depth cues with no need of the three dimensions by means of the simultaneous calculation of the stereo-pair. The method consists of the horizontal displacement of pixel according to the assigned index. If the displacement is greater for small indices, the fractal set is located in a valley and the outer regions descend towards him. If the displacement is greater for greater indices, the set is a mountain from which it is descended to remoter regions. If the images swap the opposite happens. The table 1 shows the different situations. k is the loop counter, namely, the index we use to build the displacement.

Greater displacement	Left eye	Right eye	Depth of the set (greatest k)
great k	right shift	left shift	near
great k	left shift	right shift	far
small k	right shift	left shift	far
small k	left shift	right shift	near

Table 1: Different image possibilities

The algorithm used for case #3 is the next.

Establish the correspondence Z(n, m)
For each pixel (n,m)
  Z=0+i0
  C=Z(n, m)
  k=0
  While |Z|<MaxMod & k<MaxIt
    *Z=Z*Z+C*
    k=k+1
  LeftBuffer(n+shift(k), m)=color(k)
  RightBuffer(n-shift(k), m)=color(k)
Fill up Buffers
Send LeftBuffer to left eye
Send RightBuffer to right eye

In practice, only one buffer is shifted because the resulting eye displacement is equivalent. Notice that filling up the buffer gaps is necessary because of the holes generated by the shift.

3. RESULTS

The results allow us, with low resolutions, to explore interactively the landscape generated by the fractal image in "three dimensions". The depth coordinate hasn't been computed but simulated in the way we describe above. Here we present some images obtained with the displacement method. The figure 1 shows the stereo-pair eye-crossed as usually. The images 2 and 3 show a composition of stereo-pair in a anaglyph separation technique so they have to be seen with a red-blue glasses [Stereoscopy]. A threshold module of 2 has been used for all. The maximum k value is 60. The shift function is linear in all cases.

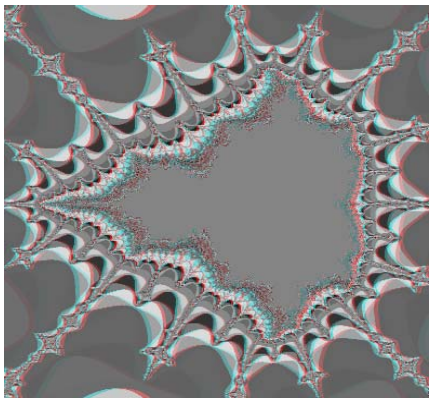


Figure 2: Anaglyph of Mandelbrot set

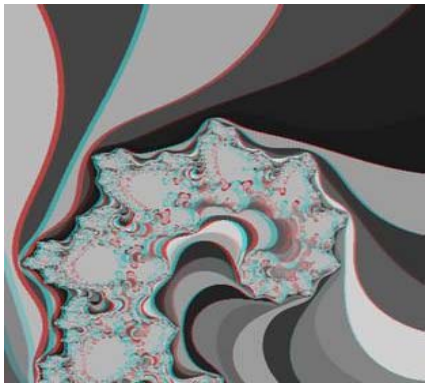


Figure 3: Anaglyph of Julia set

4. ACKNOWLEDGEMENTS

This work has been supported by projects TIC1999-0510-C02-01 and PPI-UPV2002.

5. REFERENCES

[Peitgen86] H.Peitgen and P.Richter, *The Beauty of Fractals*, Springer-Verlag, 1988.

[González02] A.Gónzalez. Personal web-page. <http://www.users.inycom.es/~agonzalez>

[XplorA02] XplorA Project, Computer Graphics Section at Polytechnic University of Valencia, <http://www.dsic.upv.es/users/sig>

[Stereoscopy] <http://www.stereoscopy.com>

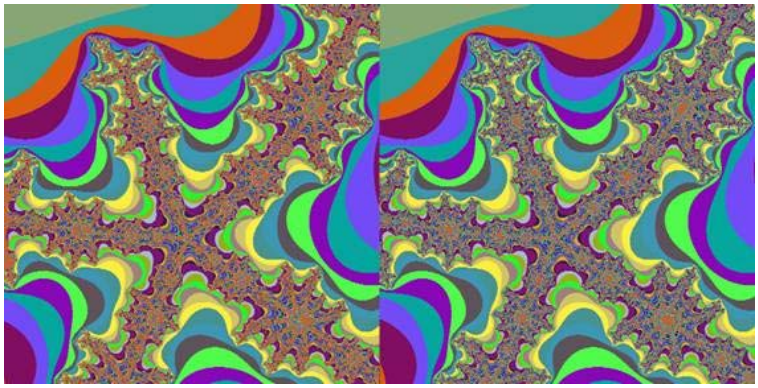


Figure 1: Stereo pair with images eye-crossed

Surface and Volume Reconstruction of the Left Ventricle from SPECT data

Oscar García Panyella
CITEM
EA La Salle / Universitat Ramon Llull
C/ Quatre Camins 2, 08022 Barcelona
oscarg@salleURL.edu

Antoni Susín Sánchez
Departament de Matemàtica Aplicada I
Universitat Politècnica de Catalunya
Avda. Diagonal 647, 08028 Barcelona
Toni.Susin@upc.es

Abstract
This document describes the three-dimensional reconstruction of the internal and external surfaces of the human's left ventricle from actual SPECT data. The reconstruction is a first process fitting in a complete VR application that will serve as an important diagnosis tool for hospitals. Beginning with the surfaces reconstruction, the application will provide volume and interactive real-time manipulation to the model. We focus on speed, precision and smoothness for the final surfaces. As long as heart diseases diagnosis requires experience, time and professional knowledge, simulation is a key-process that enlarges efficiency.

Keywords
Tele-medicine, deformable models, virtual reality, image filters, 3D surface.

1. INTRODUCTION

Access to a 3D model obtained from patient's data can have several applications like support on diagnosis, surgery planning, student's training or even remote-operation. A first approximation to the problem would be using a manual process with specific image-processing software though it would require deep medical knowledge and experience.

Some parts of the human body like skin or bones have clear intensity-gradient variations that make surface-reconstruction techniques suitable. Unfortunately these techniques don't work when regions are formed from soft tissue. That's the case of the heart, liver or muscles.

In order to solve those cases several contour-based techniques have been introduced. In all the approximations the evolution is similar, an initial parametrically defined contour or mesh which deforms attracted to a certain energy minimum, numerically implemented as a finite difference method.

Another possible approach is to use the finite element method which gives nice results but is computationally expensive and complex.

The dynamic model that we present solves the same type of problems using the evolution of a deformable mesh affected by internal and external forces. Internal forces are defined in terms of elasticity; external forces are derived from the data set as a vector field called *Gradient Vector Flow* (*GVF*) [Xu97][Xu98].

The use of the *GVF* vector field avoids some of the limitations of traditional snakes related to initial distance to data and robustness in concave regions.

2. INPUT DATA

The system takes as its input SPECT, *Single Photon Emission Computed Tomography*, images. Those images give functionality keys about the organ and do not describe its anatomy. Thus data shows the activity being held in terms of the amount of useful tissue, without giving a clue about shape. From this knowledge it becomes clear that ischemic areas, it means in absence of blood irrigation, won't be shown in the images. That's the case of ventricle areas being affected by a heart attack.

3. INTERNAL FORCES

Internal forces define four possibilities for the model:

- *Plane deformation model*, where each of the triangles in the mesh has its own elasticity forces. Deformation will be characterized by the added action of three forces: stretch, shear and bend.
- *Spring-mass deformation model*, where the only internal force is stretch, defined between pairs of particles following *Hook*'s equation.
- *Restricted spring-mass deformation model*, where spring-forces are only allowed in the direction normal to the derived vector field.
- *Free deformation model*, where the only existing force is the external one, derived from the data set. There's no connectivity between particles and topology must be maintained using a smoothing algorithm, apart from the evolution scheme.

All the models have been extensively reported in [Amatller00], [García00] and [Susin01]. The smoothing algorithm can be found at [García02].

4. EXTERNAL FORCES

External forces consist on the minimization of a functional that mixes the information derived from the image-intensities gradient with a diffusion term that allows the vector field to be spread out.

The vector field that acts as the external force is then obtained by minimizing the functional shown in figure 1.

$$\varepsilon = \iiint \mu \sum_{i \in \{x,y,z\}} (u_i^2 + v_i^2 + w_i^2) + |\nabla I|^2 |V - \nabla I|^2 \, dxdydz$$

Figure 1: The energy-based functional [Xu97][Xu98].

The functional consists on two well-differentiated terms. On the left, the diffusion term that spreads the field when variations on intensities are negligible. On the right the property term, that dominates the expression when variations are important. The μ parameter will control the balance between both terms.

5. DYNAMIC EVOLUTION

The dynamic evolution is based on *Newtonian* classics, as it is shown in figure 2:

$$F_i = m_i \cdot \ddot{x}_i \begin{cases} \dot{x} = v \\ \dot{v} = F_i / m_i \end{cases}$$

Figure 2: Newtonian dynamics.

In order to solve the system, we need an appropriated numerical method. This method or solver, depends on the deformable model applied. We have used:

- Implicit *Euler*'s method when using the plane deformable model.
- Explicit methods like *Euler*, *Midpoint* or *Runge-Kutta-4* when using the spring-mass, restricted spring-mass and free deformation models.

Reports on these can be found at [Amatller00], [García00] and [Susin01].

6. TEST MODEL: the PHANTOM

In order to measure the reliability of the system from the results point of view, it becomes necessary to test a data set with known volume. The *PHANTOM* is a test-model that offers known geometry and volume, allowing us to compare with the obtained results. In our case we have used the one from the *Vall d'Hebron* Hospital which is presented in figure 3:

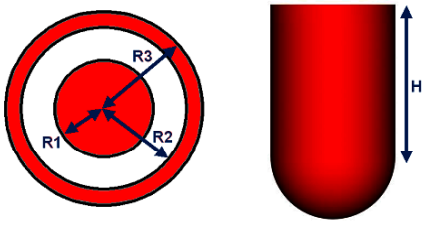


Figure 3: The PHANTOM dimensions.

With R1=20mm, R2==35mm, R3=45mm and H=55mm. Those distances give external and internal volumes of 305301.4 mm^3 and 89794 mm^3 respectively.

7. RESULTS

7.1 Oscillations

If we analyse several particles trajectories:

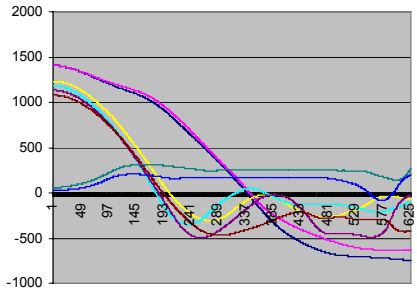


Figure 4: Particles oscillate in data frontiers.

We can see the oscillation effect in the frontiers of the data. Figure 4 shows the distance between several particles and the data to be recovered. Defining the black line (0) as the zero-distance to data, it becomes clear that particles oscillate while system iterates, even if they have reached the surroundings of the data frontiers.

This behaviour demands for a stopping mechanism that we have implemented by using an accurate segmentation filter: the *Canny-Edge Detector* [Canny86], in order to mark the border voxels (data frontiers).

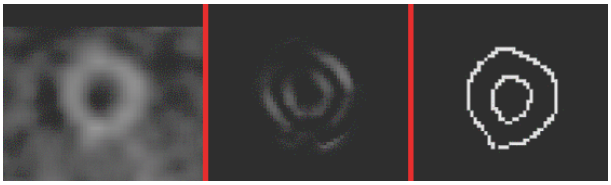


Figure 5: Original *SPECT* image (left), after *Reynolds* operator (middle) and after *Canny* operator (right).

Figure 5 shows some results. An appropriate tuning of the filter (1.8 for the Gaussian, 0.3 for the low threshold and 0.8 for the high threshold) ensures performance for *SPECT* images rather than using less sophisticated operators like the *Reynolds* gradient.

7.2 Distance to real data

We present a comparison between deformation models and final distance of the surface to data in table 1.

K and KD stand for constant and damping constant respectively.

Initial mesh consisted on 642 particles and 1280 triangles. We used a GVF balancing constant of 25.0 and a damping factor of 25%.

The implicit scheme (plane deformation) is clearly slower than the explicit one (spring-mass and free cases) but Δt can be bigger in the implicit methods. Despite we have used a smaller Δt for the explicit schemes they are faster even under those circumstances.

Note that the free model achieves 97% of the particles really close to data (less than one voxel) while the rest stays at the 75% approximately.

	Plane (1)	Plane (2)	Spring-Mass	Free
Δt	0.01 s.	0.05 s.	0.0005 s.	0.0005 s.
% < 1 voxel	77 %	74.5 %	73.6 %	97 %
K Stretch	10	10	10	---
KD Stretch	1	1	1	---
K Bend	10	10	---	---
KD Bend	1	1	---	---
K Shear	1	1	---	---
KD Shear	0.1	0.1	---	---

Table 1: Final distances depending on the deformation model.

7.3 Size of triangles

The final 3D model, in the future, will consist both on surface and on internal volume. It will be necessary to perform a tetrahedralization between both surfaces, external and internal. This process, which will be a further step in our project, will permit the real-time interaction with the synthetic organ. Tetrahedralization will associate internal triangles with its external neighbours. Then it is imperative to ensure the final quality of the mesh, rejecting all possible degenerations.

A detailed mesh will recover data with minimal error but won't be smooth enough. On the contrary, if the initial triangle size is major or equal than the separation between data slices, we can ensure a good aspect ratio for the triangles (area / major edge).

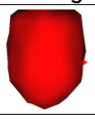
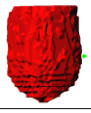
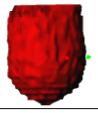
	Flat shading	Gouraud shading	Averaged	Max	Min
1			**84.03 / 5.1**	**170.02 / 7.73**	**23.15 / 1.99**
2			**22.5 / 2.57**	**116.4 / 5.24**	**0.82 / 0.09**
3			**6.24 / 1.29**	**72.38 / 4.95**	**0.0007 / 0.0005**

Table 2: Reconstructions depending on the LoD of the meshes (mm^2 / averaged).

Table 2 shows a recovered *PHANTOM* using three different meshes. First case (low LoD) does not present any degenerations. Second case (medium LoD) deals with some degenerations as we can derive from its minimal area value. Third case (high LoD) presents clear degeneration problems.

We conclude that the first mesh, where triangle dimensions are similar to space between data-slices, is smooth enough and gives not degenerations.

7.4 Volume estimation

Volume results are presented comparing the real data volume with the one reconstructed. We compute the real volume by filtering data voxels, rejecting those with intensities lower than 20% of the maximum.

If we quantify the data volume we obtain a value of 368782 mm^3. This volume differs from the theoretical one in a 12%. We must take in account the low resolution of the data and the error due to the quality of the reception from the medical equipment.

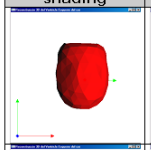
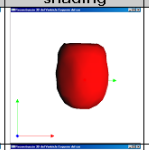
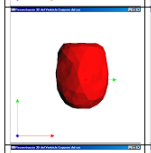
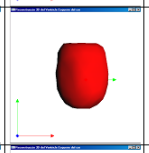
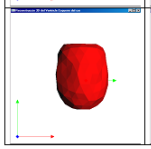
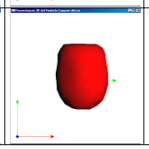
Flat shading	Gouraud shading	Error (%)	ΔT (s)	Time (s)	Solver
		1.44	0.1	0.322	Euler
		0.68	0.12	0.14	Mid Point
		1.53	0.17	0.111	RK4

Table 3: Volume results vs. numerical method.

Simulations on table 3 have been executed using the free deformation method with the smoothing algorithm active. The selected initial mesh is the one labelled as 2 in table 2. Third column shows the computed % of error between the recovered and the real data. All three numerical schemes are satisfactory achieving errors less than 2%.

7.5 A complete cardiac cycle reconstruction

Figure 6 presents a complete cardiac cycle recovered from actual patient's data. The cycle is formed by eight temporal acquisitions. Each data set consists on 64 x 64 x 24 voxels, with spatial resolutions of 2.87 mm (X), 2.87 mm (Y) and 5.74 mm (Z).

The meshes were generated using the free deformation model with the RK4 explicit scheme, using a GVF balancing constant of 10, a damping factor of 1% and a time increment Δt of 0.1 seconds.

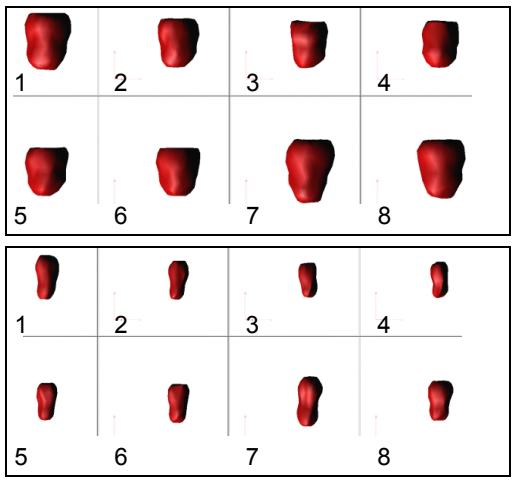


Figure 6: Complete cardiac cycle with external surfaces (top) and internal surfaces (bottom).

Table 4 presents internal, external and wall volumes in mm^3.

	1	2	3	4
Int.	74741	45677	36523	32956
Ext.	443762	316804	254112	250131
Wall	369021	271127	217589	217175
	5	**6**	**7**	**8**
Int.	47237	55825	79581	63133
Ext.	299779	345830	473608	451525
Wall	252542	290005	394027	388392

Table 4: Volumes for the eight temporal instances.

As a first diagnosis tool, physicians use the Ejection Fraction (EF) parameter. Figure 7 shows the parameter for the ventricle in figure 6:

$$EF(\%) = \frac{DiastoleEndVolume - SystoleEndVolume}{DiastoleEndVolume} x100 = 58.6\%$$

Figure 7: EF calculation equation.

Where all the volumes are internal. EF values inside the interval 50%-70% stay for a non-pathological situation.

Figure 8 shows the complete process for a reconstruction:

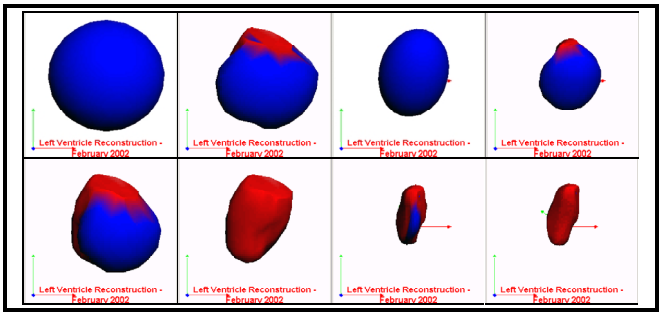


Figure 8: External (left) and internal meshes (right).

The sequence shows first the initial meshes in blue. The reconstruction begins and the meshes turn red as long as the particles reach the marked voxels.

7.6 Reconstructions with missing-data

When a patient has had a heart attack some of the areas of his heart become ischemic and, because of the absence of blood irrigation, the data we obtain can have some missing zones. We made some recovering test experiments with 10%, 32% and 53% percentages of missing volume data, always referred to the 100% of the total *PHANTOM* volume.

In these tests, a different *PHANTOM* model (more geometric heart-like) was used. Its volume was 265501 mm^3. Table 4 shows the experiment.

Here the first column shows the initial data to be recovered, the second column depicts the final meshes and the third column points out the recovered final volumes (absolute values and percentage of total).

As we can see, the best recovering is the third one because it gives a percentage of missing volume of 29.1% against the 32% of the real data emptied. This represents a relative percentage error of 0.09. For the other test examples we obtain relative errors of 0.45 and 0.24 respectively.

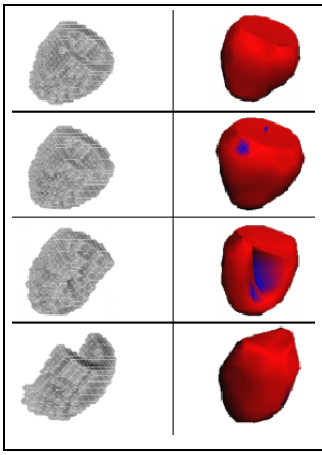
	Volume=265501 mm^3 % of total=100% Volume=250944 mm^3 % of total=94.5% Volume=188360 mm^3 % of total=70.9% Volume=159970 mm^3 % of total=60.2%

Table 4: Recovered surfaces from partial data.

8. CONCLUSIONS

We have presented the results obtained so far with *SPECT* data. In terms of precision, computing time and model accuracy, different strategies have been tested.

The final result is accurate enough for the usual medical practice. A compromise between speed and precision has to be assumed. In our present situation, we have decided to use the fastest method after testing the other approaches and checking that acceptable precision is achieved. As a function of the final application other strategies can be chosen.

9. REFERENCES

[Amatller00] J. Amatller, O. García, A. Susín. "Modelo Dinámico para la segmentación automática de imágenes 3D", X Congreso Español de Informática Gráfica-CEIG'2000, pág. 355-370, Junio 2000.

[Canny86] J. Canny. "A Computational Approach to Edge Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 8, No. 6, Nov. 1986.

[García00] O. García, A. Susín, I. Navazo. "Segmentación Automática mediante un modelo dinámico. Aplicación a la reconstrucción del ventrículo izquierdo", Jornadas de Investigación en Ingeniería Biomédica, Sitges, 2-3 Noviembre 2000.

[García02] O. García i Panyella. "Diploma d'Estudis Avançats", Escola Tècnica Superior d'Enginyeria Electrònica i Informàtica La Salle, Universitat Ramon Llull. Març 2002.

[Susin01] A. Susín, O. García. "Modelo dinámico para la reconstrucción del corazón humano", CEDYA 2001, XVII Congreso de Ecuaciones Diferenciales y Aplicaciones y VII Congreso de Matemática Aplicada, Universidad de Salamanca, 24-28 de Septiembre de 2001.

[Xu97] C. Xu and J.L. Prince, "Gradient Vector Flow: A New External Force for Snakes", IEEE Proc. CVPR, 1997.

[Xu98] C. Xu and J.L. Prince, "Snakes, Shapes, and Gradient Vector Flow", IEEE Transactions on Image Processing, pag. 359-369, 1998.

Visualization of values of solar radiation using geomatic tools

Francisco Feito Higueruela
Departamento de Informática,
Escuela Politécnica Superior,
Universidad de Jaén, 23071
Jaén, España.
ffeito@ujaen.es

Antonio J. Gil Cruz
Mª Isabel Ramos Galán
Departamento de Ing. Cartográfica, Geodésica y Fotogrametría
Escuela Politécnica Superior
Universidad de Jaén, 23071
Jaén, España
{ajgil, miramos}@ujaen.es

David Pozo Vázquez
Joaquín Tovar Pescador
Departamento de Física Aplicada
Escuela Politécnica Superior
Universidad de Jaén, 23071
Jaén, España
{dpozo, jtovar}@ujaen.es

Abstract
In this work we present a preliminary study of an environmental phenomenon, solar irradiance on ground level, using data from digital elevation models (DEM), meteorological satellites, GPS, radiometers, etc., emphasizing the utility that has methods of scientific visualization and interpretation as well as the application and management of the data for the analysis and resolution of incidents where they take part this type of phenomena.

Keywords
Scientific visualization, environment, modelling, DEM, solar radiation, meteorological satellites, radiometers, GIS, GPS.

1. INTRODUCTION

In the last years has increased considerably the necessity to have a greater knowledge of aspects related to Sciences of the Earth. Within these disciplines is beginning to develop a new investigation that tries to analyse the different sources from information: digital elevations models of the land (DEM), remote sensing, GPS, meteorological stations etc. along with visualization methods, interpretation, application and management of the data to solve environmental problems.

The main point of this work is the knowledge of the global radiation in surface since it is of great interest its application in as diverse scopes as the hydrology, the climatology, the study of the biological processes, applications related to the electrical energy or agriculture.

2. WHY DO WE MODEL SOLAR RADIATION?

Some networks of meteorological stations exists, they measure solar radiation continuously, nevertheless, these are few and they are very dispersed. The analysis of values of radiation for hydrologic, climatologic, agricultural, biological studies, etc..., requires the taking radiation measures on ample regions of continuous way in the time, and with a high space resolution. Also it demands long and expensive campaigns of data measurements in the field with results, sometimes, insufficient.

2.1 Habitually used method

To estimate solar radiation in surface, measures in the land are taken by radiometers in disperse. To obtain values of radiation in points different from the measured ones is made a interpolation / extrapolation of these values to obtain a lot of points of greater or smaller density according to what we determine. This methodology gives trustworthy results in those places where the variability of the values of Radiation is not very great, but it is inadequate for those zones of great topographic complexity where the radiative behaviour is very variable.

In these cases the estimation of the irradiance in surface could be solved by means of radiometric networks of high cost and difficult maintenance, although it would not either be reached in many cases the correct results since, as we aimed more ahead, we have to consider the topography of the zone.

2.2 Solar radiation and topography

Under cloudless sky or totally covered the radiative variability depends fundamentally on the topography. The variations in the slope, the type of surface, the reflectance and the obstructions of the land affect to the solar radiation in surface.

In zones of great topographic variety the space autocorrelation of solar radiation varies between 300 and 1000 W/m^2 for the most of lands. For this reason it is difficult that a network of stations on ground level could catch the space variability that the solar radiation presents. For that reason we need a system and a methodology able to interrelate several phenomena to each other considering the effect that produces one on the other showing in a while the joint result of that interrelation of elements: sun, slopes, altitudes, reflectance, etc.

2.3 Methodology to estimate a solar radiation model

We present an initial solution to the problematic, a method to obtain values of radiation in zones with complex topography by means of the use of satellite images and digital elevation models (DEM). The study has been made on a zone of 15 x 20 km corresponding to the Natural Park of Sierra Mágina (Jaén), where heights are registered on the mean sea level from 520m to 2160m. we have used a DEM corresponding to the province of Jaén yielded by the Consejería de Medioambiente of the Junta de Andalucía, it has a resolution of 30 x 30m interpolated to 20 x 20 m. The DEM provides information to us on the geometry of the surface in relation to the Sun.

A phase of the work consists of obtaining data of solar radiation from measures of satellites METEOSAT and NOAA, sensor AVHRR [Möser84], [Cano86], [Stuhlmann90], [Laszlo93], [Beyer95], [Beyer96]. The combined use of both satellites displays one double advantage: on the one hand, the use of satellite data from stationary satellites as METEOSAT that provide a angle of fix view for a determine region and greater frequency of images (until 10 images by day surroundings at noon premises), on the other hand, offer the advantage of use also satellite data de satélites polestar, like NOAA, that present a greater resolution space (1,1 km in front to the 5,5 km of the METEOSAT) although have minor resolution temporary.

The values of horizontal global radiation have been estimated from the models of radiative transference where we have considered the corrected optical mass in height for each point of the DEM which X, Y coordinates are known. The value of the normal irradiance obtained in surface after being attenuated due to the absorption processes and dispersion, with a propagation means homogenous supposed, comes from the expression:

$$I_{n\lambda} = I_{on\lambda}\, e^{-k_\lambda \int_0^s \rho(s)ds} = I_{on\lambda}\, e^{-k_\lambda m}$$

that constitutes the Law of Beer-Bourger-Lambert. This law allows to consider the radiance in a point based on the radiace incidented, the optical mass and the coefficient of extinction of means. The total attenuation for a certain wavelength due to the components of the atmosphere:

$$I_{n\lambda} = I_{on\lambda}\, e^{-k_{1\lambda}m_1} e^{-k_{2\lambda}m_2} \ldots e^{-k_{n\lambda}m_n} = I_{on\lambda} \prod_{i=1}^{n} e^{-k_{i\lambda}m_i} =$$

$$= I_{on\lambda}\, e^{-\sum_{i=1}^{n} k_i m_i} = I_{on\lambda} e^{-k_\lambda m}$$

Defining the transmitance for a certain wavelength of the atmospheric components, now the previous expression is:

$$\tau_\lambda = \frac{I_{n\lambda}}{I_{on\lambda}} = \prod_{i=1}^{i=n} \tau_{i\lambda} \qquad I_{n\lambda} = I_{on\lambda} \prod_{i=1}^{i=n} \tau_{i\lambda} = I_{on\lambda}\tau_\lambda$$

The value of the direct irradiance in a part or all the spectre of the solar radiation is obtained integrating the direct spectral irradiance for the corresponding interval of wavelengths.

$$I_n = \sum_0^\infty I_{n\lambda} \Delta\lambda$$

Expressing I_n based on the extraterrestrial value and on the transmitances of each one of the components of the atmosphere:

$$I_n = \sum_0^\infty \left(I_{on\lambda} \prod_{i=1}^{i=j} \tau_{i\lambda} \right) \Delta\lambda$$

and based on the transmitances:

$$I_n = \sum_o^\infty (I_{on\lambda}\tau_\lambda) \Delta\lambda$$

If it is desired to obtain the value of the direct irradiance on a horizontal surface (I_h), is due to project on this surface.

3. TOOLS FOR MODEL VISUALIZATION

We can not study any environmental phenomena without having a model that show us the behaviour of it. So, we need some tools to visualise solar radiation and to analyse the values in some areas and the difference from one zone to another. The visualization process is made up of the following phases:

3.1 Necessity of a system to manage data

We have established a grid that covers the zone (20 km x 15 km) so that each cell, of 20 m x 20 m, is going to have an estimated value of radiation. This estimated value either comes from the DEM or from the satellite images of different space-temporary resolution.

To complete the analysis of variability of radiation is useful to include data taken on the ground with radiometric sensors. GPS system will be used to georreference points and to include them in the model already created.

We have data coming from different sources: DEM, satellites images, data from GPS, radiometers, etc... all information needs a management system that allows to store it, order it properly and classify it so that it be able to accede to it at any moment, also to be possible to work indifferently with any data set in simultaneous or

alternative way obtaining results as fast as possible. We have numeric data (values each 20 m x 20 m): coordinates, solar radiation values and altitude values above sea level, with a properly data base it is possible to add values of temperature, humidity, precipitation, vegetation and so many layers of information as we considered opportune to enrich our environmental study. All this information is associate to a concrete zone of the land, so each point with well-known coordinates X, and or Latitude, Length would display a concrete value of solar radiation, altitude, temperature... Therefore, we have spatial data (points or zones on the land) to which we have assigned numerical or alphanumeric values (attributes) of diverse nature. How are we going to manage all these data? by means of a Geographical Information System (GIS). In our case the tool we've used is MapInfo [Daniel99].

3.2 Integration of data in a GIS

How do we have integrated data in a Geographical Information System?, we explain it in next paragraphs:

3.2.1 Data processing

Often, data we must integrate in a GIS are poorly processed or structured when do not exist a concrete source in which data are perfectly ordered and classified based on his geographic property, temporary, typology, etc. or the classification of the information is not fine because of the raised work, reason why is necessary to make an analysis and reconstruction of data:

a. DEM

We have a DEM of all the province, so we have to delimit our zone of study. The complete file has 2,000,000 points, which makes difficult to the system to manage such volume of data. We have to select our area of interest within the complete DEM which causes that it needs an optimal processing not to overload the system. It is more appropriate to work on the numeric data that on the graphs since these last ones are going to need more memory to carry out the processes. We have exported the file to an application of management of data base, Microsoft Access and have made the selection by means of a SQL query. Later we have exported the file of the selection to a software GIS, MapInfo, to generate for that new file its spatial information associated (points with known coordinates X, Y). In this way it is delimited our DEM corresponding to our zone of interest and that now consists of 787500 points, it bases far below to the previous one, this new one will allow us to work more efficiently.

b. Solar radiation values.

We applied the equation seen in previous sections to obtain the radiation values from the heights to obtained an image of 15 km x 20 km of identical resolution that the DEM (20 m x 20 m).

We are interested in analysing the image of radiation for different resolutions so that we could compare these values with the satellite's NOAA and METEOSAT ones, for that reason, we have degraded the image (with a program of treatment of images, ERDAS) from 20 m x 20 m resolution to inferior resolutions 1,1 km and 5,5 km respectively. We have obtained the following results:

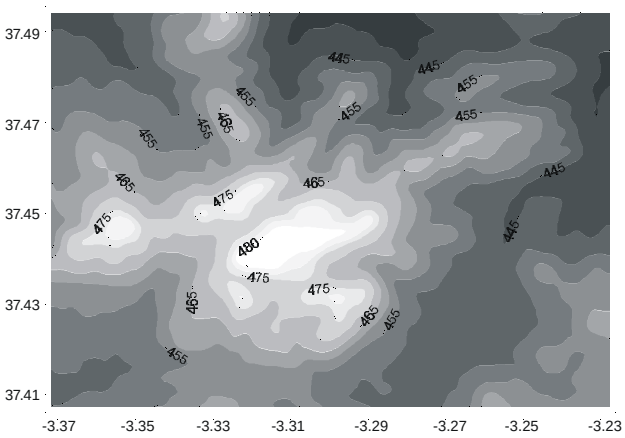


Fig 1.- Digital Elevation Model

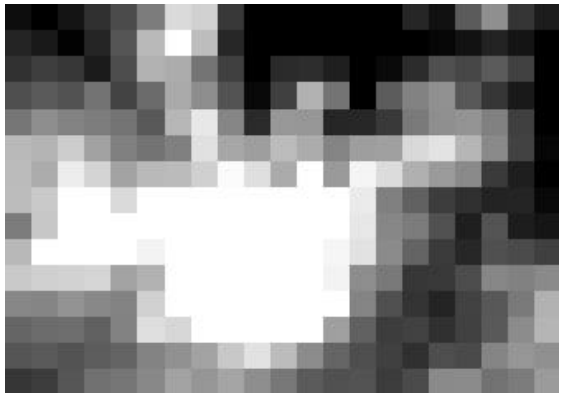


Fig 2.- NOAA´s image (1.1x1.1 km)

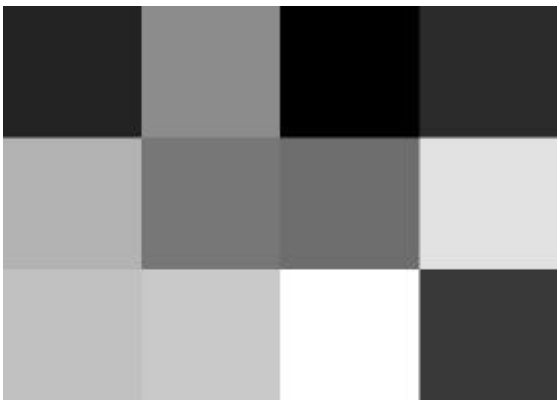


Fig 3.- METEOSAT´s image (5.5x5.5 km)

In a GIS these images are stored as tables whose spatial or graphical information are points (since in our case we are working with a vectorial GIS) whose attributes are radiation values of each one.

c. GPS

We have an estimated model of radiation values. This model must be checked by specialists in this phenomenon to analyse to what extent this model adjusts to the reality. It is for that reason that will be necessary to check these estimated values with measures taken in field.

In the first place we have to select on the zone the most suitable points to compare estimated and measured value: points located in a zone of strong unevenness or an

inflexion point of the model that could cause a strong variation on it if its estimated value is not correct, secondly also it is interesting to take values in field to enrich the model and make a joint adjustment of estimated and measured values. GPS system is going to allow to us to georreference the points in which radiation measures are taken. This tool is very useful because we can simultaneously use it with the radiometric sensors obtaining values of radiation and the coordinates of each point to the same time.

3.2.2 Visualization of the information

The potentiality of the model of radiation is increased if it is able to responds to diverse queries done by different disciplines. Thus for example, a professional of agriculture will need to know how she is going to repel those solar values on the harvests or for a professional of hydrology it will be important to see how she affects streams, and so on. The model must be able to answer those consultations so it is necessary to have data enough or the necessary layers of information so that GIS was able to interrelate all data.

Values of solar radiation on the studied zone acquire interest when we have tools to be able to visualise its spatial distribution and thus to evaluate its incidence on the elements to which it can affect. To compare obtained values of radiation with the elevations of the land it can be useful to use graphical in 3D. For the DEM we can use a range of colours where the brown ones, in different tonalities, correspond to the zones of smaller level and the green ones to the highest points, giving therefore an image similar to the real one where the mounts are going away to seat in high summits. For the graph of space distribution of radiation´ values, warmed tonalities have been adopted, as yellows and oranges to represent the maximum and minimum values.

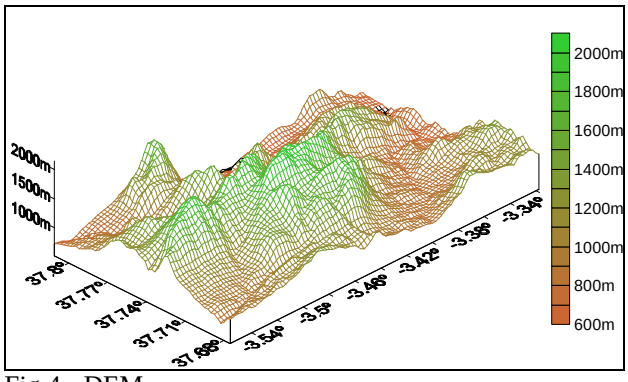


Fig 4.- DEM

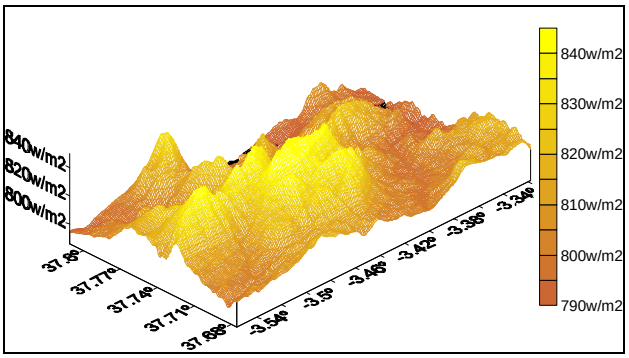


Fig 5.- Solar radiation Values

These elevations´ and radiation´ data have been integrated in a GIS along with other layers of information corresponding to the zone of study. This allows us to mix all the information that we arrange from the Information System and it also offers therefore the possibility of studying this phenomenon in depth, analysing the different factors to which it can affect and the correlation with other phenomena or possible existing anomalies in the region.

3.3 References

[Beyer95] Beyer, H.G., Constanzo C. and D. Heinemann (1995). Modifications of the Heliosat procedure for irradiance estimates from satellite images. Solar Energy 56, 207-213.

[Beyer96] Beyer, H.G., Constanzo C. and Reise Ch. (1995). Multiresolution analysis of satellite-derived irradiance maps an evaluation of a new tool for the spatial characterization of hourly irradiance fields. Solar Energy 55, 9-20.

[Cano86] Cano, D., Monglet, J.M., Albuisson, M., Guillard, H. (1986). A method for determination of the global solar radiation from meteorological satellite data. Solar Energy, 37, 31-39.

[Daniel99] Daniel L., Loree P. y Whitener A. (1998), INSIDE MapInfo Professional. ONWORD PRESS.

[Laszlo93] Laszlo I. and Pinker R.T. (1993). Global distribution of surface solar irradiance as observed from satellites. Proc. ISES Solar World Cong., Budapest 2, 179-184.

[Möser84] Möser, W. and Raschke, E. (1984). Incident solar radiation over Europe estimated from METEOSAT data. Journal of Climate and Applied Metereology, 23, 166-170.

[Stuhlmann90] Stuhlmann, R., Rieland, M., and Raschke, E. (1990) An improvement of the IGMK model to derive total and diffuse solar radiation at the surface from satellite data. Journal of Applied Meteorology, 29, 586-603.

Voxel Matching Reconstruction in Real Image Sequences of Human Avatars

Jose Maria Buades Rubio
Computer Graphics and Vision Group
Department of Computer Science
Universitat de les Illes Balears
josemaria.buades@uib.es

Francisco Jose Perales Lopez
Computer Graphics and Vision Group
Department of Computer Science
Universitat de les Illes Balears
paco.perales@uib.es

Abstract

This article presents a part of a system that is used to analyse and synthesize human movement by means of a color segmentation and matching process to track and reconstruct the main aspects using a biomechanical model of a person. In particular, we explain the application of the Marching Cubes algorithm obtained from a set of voxel information. A color segmentation criterion is proposed. The segmented parts help us reduce the space search in the matching process. The main purpose of the system is to carry out a correspondence between this graphic model and the person in movement in several real images.

Keywords

Human Motion Analysis, Background Substraction, Voxel Representation, Marching Cubes.

1. INTRODUCTION

The analysis of the movement of the human body may be approached from different perspectives, depending on the type of application to be considered. In our case, the techniques used are conditioned according to the initial hypotheses of minimum perturbation of movement and/or surroundings, using exclusively visual information of the scene and a biomechanical and graphic model of the person.

With this aim in mind, in order to obtain the parameters of human movement we use computer vision techniques (pre-processing, color segmentation, matching of entities, camera calibration, 3D reconstruction etc.), capturing the individual from an arbitrary number of color and grey level video cameras which enable us to obtain as much information as possible.

This article presents a part of a system that is used to analyse and synthesize human movement by means of a color segmentation and matching process to track and reconstruct the main aspects using a biomechanical model of a person. In particular, we explain the application of the Marching Cubes algorithm [Lorensen87] obtained from a set of voxel information. A color segmentation criterion is proposed. The segmented parts help us reduce the space search in the matching process. The main purpose of the system is to carry out a correspondence between this graphic model (primitives) and the person in movement in several real images (color and grey level).

The original voxel approximation [Lorensen87] is used to roughly fit the volume occupied by the person in the scene. The process is intended to be non-invasive and automatic, although it is currently used with the minimum manual intervention of the user. The system works in a controlled environment. The final result will enable us to integrate the synthetic model and its movement with the real person in a real or virtual world adaptable to different applications.

2. CAPTURING PROCESS

In the capturing process we have two possibilities; to use two synchronized color cameras, or four synchronized black and white ones. In this case we use only the two interlaced color cameras, because the voxel algorithm is based on color information.

As far as the calibration of the cameras is concerned, we used a basic algorithm which takes into account all the usual intrinsic and extrinsic parameters, although initially we are not dealing with any kind of distortion.

3. MATCHING CRITERIA

The matching process associates each articulation with a 2D point in the image and consequently a 3D projected primitive with a 2D region of pixels. This process consists of analysing each image obtained from the cameras in an instant of time *t*. Once we have the articulation located in two or more cameras, we estimate the most accurate 3D point; an articulation may only be detected in one or no image so the process will have to be completed with contextual information from a much higher level.

What is intended is to obtain the most accurate approximation possible, thus the virtual human must have similar anthropometric sizes to a person. For the measurement, the person takes up four classic ergonomic postures which enable us to obtain the maximum information.

On analysing the sequence, we are able to apply physical and temporal restrictions which help us carry out the

matching whilst reducing the errors and the search space [Perales94]. This adjustment process is conditioned by a set of conditions which are optimised in each case and type of movement. The restrictions are:

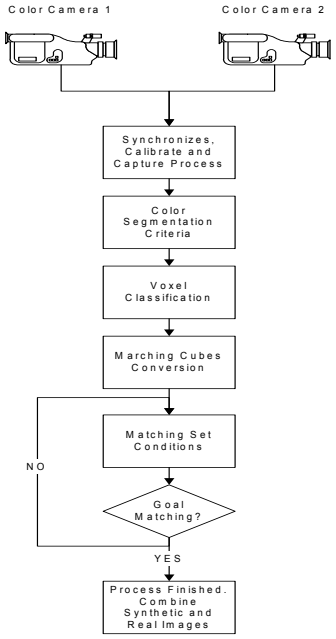


Fig 1. Process Diagram, from capture process to final results

Inclusion Condition: For all the segments of the 3D model projected onto the 2D sequences in all views, it is necessary that all segments or primitives are within the human body area obtained in the segmentation process. If the number of matched joints exceeds a predefined threshold, the condition is satisfactory.

Distance Condition: The segments of the body part should be centred in the boundary of segmented body part. This condition is used to adjust the position of the segment. Some errors are possible depending on the graphical primitive used (superellipsoids, generalised cylinders, etc.).

Position Condition: All the joints and segments should satisfy the physical conditions imposed by the biomechanical model. The algorithms proposed only search for the range within the physical angle constrains.

Temporal Condition: The position of the same joint or segment matched should be within a small distance from the position in the preceding frame. The distance for this condition is specified for a particular kind of movement.

Collision condition: There should be a one-to-one correspondence between nodes and segments and their positions in the 3D-space. If more than one primitive has the same 3D position, a collision is detected in the matching process. A match should not have collision.

Variation Object-Model Condition: We compute the variations between consecutive images and models to detect the joint variations. The idea is to reduce the number of segments updated in every iteration in the matching process.

This set of restrictions defines a level of matching. This matching process currently works in an automatic way for simple kinds of movements without multiple occlusions and no changes of topology.

This process is complex and time consuming so, we need to reduce the space search, using a voxel approximation where we suspect that the person is moving. This means that before using the set of conditions proposed, we need a segmentation process to detect occupied voxels and assign them to a part of the body.

For the time being, the automatic process detects the occupied volume, computed from the *n* different cameras, this is done in real time and at rate of 30 fps (NTSC). To compute it we carry out the following steps:

The calibrated volume $A \subset \Re^3$ is partitioned in a subset of volumes $\mathbf{X} = \{A_1, A_2, A_3, \ldots, A_N\}$ / $\forall i$ A_i is a cube of size *l*, $\cup A_i \supseteq A$ and $A_i \cap A_j = \varnothing$ if $i \neq j$
$\forall i$ compute occupied(A_i).

Algorithm 1. Compute first frame, all volume

This process is carried out for each captured frame, and for the total volume, but normally the volume the subject occupies is less than the total volume, therefore we can decrease computing time if we limit the study volume to a reduced volume. For this, if in the previous frame any voxel has been detected to be occupied, the algorithm computes the bounding box that contains the occupied volume, and only computes the bounding box for this subset volume. We use the following algorithm:

Compute the study limits in previous frame
For each voxel in bounding box
    Compute whether the voxel is occupied or not
While any of the bounding box border has any occupied voxel, and is not a total volume border
    Expand the bounding box one unit in the occupied border
    For each new added voxel
        Compute whether the voxel is occupied or not

Algorithm 2. Compute Next Frame

This allows us to restrict the studied volume to the zone of interest and to modify and move the bounding box according to the movements of the subject in relation with condition variation.

So far we have explained how to compute the volume in general, but we have not explained how to determine whether a voxel is occupied or not. A voxel is occupied if the voxel has changed in all cameras, for this process we use a background image that will serve as a reference, with this reference image and the captured frame we carry out a subtraction and apply an adaptive threshold. We then have to discern two cases: color cameras and black and white cameras. In the b/w cameras the color space is one dimensional - the light intensity captured from the camera - therefore we have little useful information. In this case we only can apply one difference.

$$abs(frame(x,y) - background(x,y)) \geq threshold \qquad (1)$$

This causes the appearance of non desired-shadows if we have a low threshold. And we will not detect interesting

parts if the threshold is high. In color cameras, the color spaces are 3D, thus we have more information that we can use. After carrying out some tests with different color spaces we finally chose the HSI color space, rather than using YUV, RGB or nRGB. Now we have a typical threshold for each component H, S and I. This color space allows us to eliminate shadows caused by the subject and retrieve information about zones, which was not possible in b/w.

To select a voxel as changed from a camera we use an independent threshold for each component and in at least one of them there should be a change higher than the threshold. The threshold has been chosen heuristically. The Hue component is cyclic, so the distance in H between 0 and 359 is 1 degree, for this reason we have to check for component H.

$$\begin{aligned} abs(frame\_I(x,y) - background\_I(x,y)) &\geq threshold\_I \\ abs(frame\_S(x,y) - background\_S(x,y)) &\geq threshold\_S \\ abs(frame\_H(x,y) - background\_H(x,y)) &\geq threshold\_H \\ abs(frame\_H(x,y) - background\_H(x,y)) &\geq 360 - threshold\_H \end{aligned} \quad (2)$$

To achieve the best results, we previously smooth the captured and background images, which gives as a result the stabilization of the H-component and therefore divides the threshold of such component by two. For the I-component, we can allocate a high threshold and thus eliminate shadows without erasing parts of interest of the subject. The task of matching now focuses on determining out of the segmented images and the generated volume which parts are of interest.

Our proposed system is related to the work presented in [Arita00] but using more views, color and specific conditions. We also reduce the shape parameters and do not consider deformation. We are working to combine the process presented in the first part of this section with the last segmentation and voxel occupancy criteria. We know that recovering the structure of the body from the voxel representation is very difficult or may be impossible. We only use the voxel representation to fit the space and to help the matching process that uses simple predefined shapes. By selecting the end effectors such as hands, feet and head and later applying inverse kinematics, we can achieve good results.

A few words are added here to explain briefly the 3D representation used. When we edit a humanoid, it is very helpful to have the person we are using as a reference and vary the measurements of the humanoid actively, therefore we have placed the captured image as a background. We create a 3D representation of the humanoid with the captured image placed as background, using OpenInventor. As the definition of the objects and their parameters in VRML are very similar to OpenInventor, the conversion from the H-Anim humanoid is very easy.

In the general automatic process, still in progress, the representation of the computed volume was done at the beginning by drawing the face of each one of the border voxels between the occupied and non-occupied volume.

4. RESULTS

In this section, we present some examples of original sequence color images, segmented images, voxel overlapped approximation and marching cube cases. Results are illustrated in several figures. See figure 2.

5. CONCLUSIONS AND FUTURE WORK

The system presented can analyse and match a segmented person with a biomechanical model in an automatic way. Finally, we can generate a virtual human in a compliant VRML format with the same measurements as the subject, for later integration in the virtual world. In order to carry out this task, we capture the subject doing a motion and the matching is carried out between the human and the humanoid. In this matching process the captured images are used as a background image and the humanoid is overlapped to verify the humanoid correct posture using the conditions considered.

Once we have reconstructed the human motion, the captured image can be erased and the humanoid inserted in the virtual world, going through the same motion.

As yet, the matching process proposed does not consider the facial features or the fingers of the person recorded. In the near future we plan to develop a high level graph model including main terminal nodes (head, hands and feet). Some important restrictions will be included when many changes of topology are presented. Also, to track hands and face we are testing an algorithm via particle filter [Isard98] which runs well with rigid regions and recovers partial occlusions.

On the other hand, the automatic process carries out the segmentation in real time with a low computing cost and can reconstruct the occupied volume. In the area of computer vision, we are trying to apply this system to recognise and track persons in a controlled environment.

6. ACKNOWLEDGEMENTS

This project is subsidized by CICYT TIC2001-0931 and HUMODAN UE project IST Program

7. REFERENCES

[Arita00] T. Nunomaki, S. Yonemoto, D. Arita, R. Taniguchi, “Multipart Non-Rigid Object Tracking Based on Time Model-Space Gradients”, AMDO 2000. Palma de Mallorca, 2000. pp 72-82.

[Gravila96] D.M. Gravila, L.S. Davis. “3-D model-based tracking of humans in action: a multi-view approach”, CVPR 1996, pp 73-80.

[HAnim] H-Anim 1.1 compliant VRML. http://www.hanim.org

[Isard98] M. Isard, A. Blake "CONDENSATION-Conditional Density Propagation for Visual Tracking" International Journal of Computer Vision 29(1),5-28 (1998)

[Lorensen87] William E. Lorensen and Harvey E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm", Computer Graphics. Proceedings of SIGGRAPH'87, Vol. 21, No 4, pp 163-169

[Nielson91] Gregory M. Nielson, Bernd Hamann, "The Asymptotic Decider: Resolving the Ambiguity in Marching

Cubes" Proceedings of Visualization'91 IEEE Computer Society Press pp 83-90

[Perales94] F.J. Perales, J. Torres. “A system for human motion matching between synthetic and real images based on a biomechanical graphical model”, IEEE Workshop on Motion of Non-Rigid and Articulated Objects, 1994, Texas.

[Perez94] F. Perez, C. Koch. “Toward Color Image Segmentation in Analog VLSI: Algorithm and Hardware”, International Journal of Computer Vision, 12:1, pp 17-42, 1994

[Wren96] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland. “Pfinder: Real-Time Tracking of the Human Body”. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 19, no 7, pp 780-785

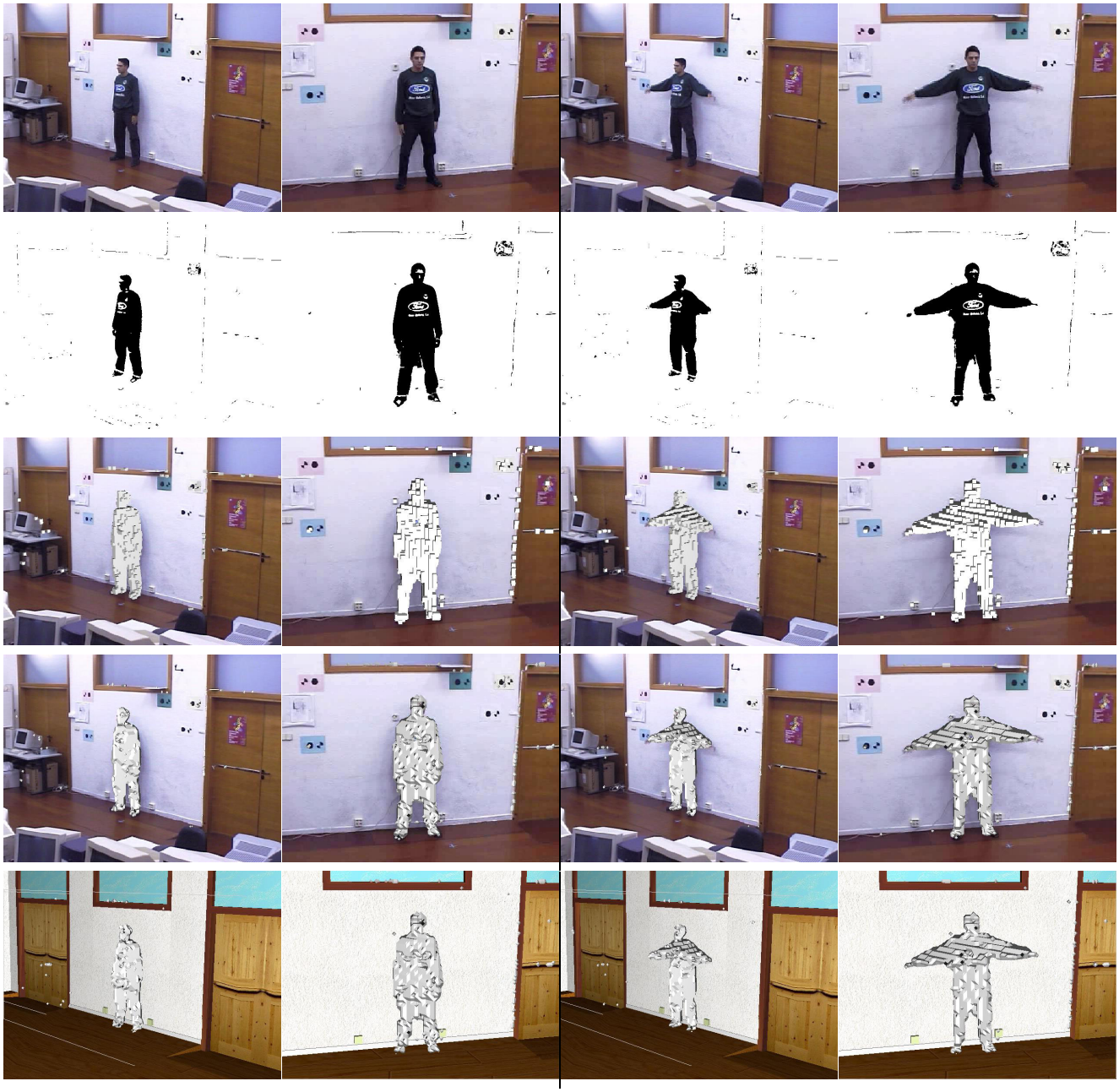


Fig 2. Two frames, left frame number 190, right frame 250 from two color cameras. Top to bottom: real image, segmentation, voxel representation, marching cubes rendered image, and virtual world mixed marching cubes

Authors Index