



HAL
open science

3D User Interfaces, from Mobile Devices to Immersive Virtual Environments

Martin Hachet

► **To cite this version:**

Martin Hachet. 3D User Interfaces, from Mobile Devices to Immersive Virtual Environments. Human-Computer Interaction [cs.HC]. Université Sciences et Technologies - Bordeaux I, 2010. tel-00576663

HAL Id: tel-00576663

<https://theses.hal.science/tel-00576663>

Submitted on 15 Mar 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Habilitation à Diriger des Recherches

Présentée à

L'UNIVERSITÉ BORDEAUX 1

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET D'INFORMATIQUE

Par **Martin Hachet**

3D User Interfaces, from Mobile Devices to Immersive Virtual Environments

Soutenu le : 3 Décembre

Après avis des rapporteurs :

Patrick Baudisch	Hasso Plattner Institute
Doug Bowman	Virginia Tech
George Drettakis	INRIA

Devant la commission d'examen composée de :

Patrick Baudisch	Hasso Plattner Institute	Rapporteur
Doug Bowman	Virginia Tech	Rapporteur
Myriam Desainte-Catherine	ENSEIRB	Examineur
George Drettakis	INRIA	Rapporteur
Pascal Guitton	Université de Bordeaux	Examineur
Guy Melançon	Université de Bordeaux	Examineur

Abstract / Résumé

Enhancing interaction between users and 3D environments is a challenging research question that is fundamental for the positive widespread of interactive 3D graphics in many fields of our societies (e.g. education and art). In this document, I present various 3D User Interfaces (3D UIs) we have developed these past few years, and that contribute to this general quest. The first chapter focuses on 3D interaction for mobile devices. In particular, I present 3D UIs for interaction from keystrokes, and for interaction from stylus/finger input. Then, I present two multi degrees-of-freedom prototypes based on the embedded cameras of mobile devices. In the second chapter, I concentrate on 3D interaction for general touch-screens (e.g. tables and large interactive screens). I present Navidget, an example of 3D UI dedicated to camera viewpoint control from pen/finger inputs, and I discuss the challenges of 3D interaction on multi-touch devices. Finally, the third chapter of this document is dedicated to immersive virtual environments, with a strong focus on musical interfaces. I present the new directions we have explored to enhance interaction between musicians, audiences, sounds, and interactive 3D environments. I conclude by discussing some directions for the future of 3D User Interfaces.

Keywords: 3D user interfaces, 3D interaction, mobile devices, touch-screens, multi-touch, musical interfaces, virtual reality, user studies.

Résumé en français

Améliorer l'interaction entre un utilisateur et un environnement 3D est un défi de recherche primordial pour le développement positif des technologies 3D interactives dans de nombreux domaines de nos sociétés, comme l'éducation. Dans ce document, je présente des interfaces utilisateur 3D que nous avons développées et qui contribuent à cette quête générale. Le premier chapitre se concentre sur l'interaction 3D pour des terminaux mobiles. En particulier, je présente des techniques dédiées à l'interaction à partir de touches, et à partir de gestes sur les écrans tactiles des terminaux mobiles. Puis, je présente deux prototypes à plusieurs degrés de liberté basés sur l'utilisation de flux vidéos. Dans le deuxième chapitre, je me concentre sur l'interaction 3D avec les écrans tactiles en général (tables, écrans interactifs). Je présente Navidget, un exemple de technique d'interaction dédié au contrôle de la caméra virtuelle à partir de gestes 2D, et je discute des défis de l'interaction 3D sur des écrans multi-points. Finalement, le troisième chapitre de ce document est dédié aux environnements virtuels immersifs, avec une coloration spéciale vers les interfaces musicales. Je présente les nouvelles directions que nous avons explorées pour améliorer l'interaction entre des musiciens, le public, le son, et les environnements 3D interactifs. Je conclus en discutant du futur des interfaces utilisateur 3D.

Mots clés: Interaction 3D, terminaux mobiles, écrans tactiles, multi-points, interfaces musicales, réalité virtuelle, études utilisateur.



Contents

Curriculum Vitae	vii
Introduction	1
1 Mobile 3DUI	7
1.1 Introduction	7
1.2 Interaction from keystrokes	8
1.3 Interaction from the mobile device touch-screens	11
1.4 New inputs for mobile devices	14
1.5 Conclusion	18
2 Touch-based Interaction	21
2.1 Introduction	21
2.2 Navidget	22
2.3 Towards multi-touch interaction with 3D data	24
2.4 Conclusion	28
3 Immersive Interfaces	31
3.1 Introduction	31
3.2 Input devices	31
3.3 Image-Sound-Human interaction	34
3.4 Conclusion	38
4 Perspectives and Conclusion	39
4.1 Future work	39
4.2 Conclusion	42
Bibliography	44
Appendix	51

Curriculum Vitae

Personal data sheet and current position // Etat civil et professionnel

Dr. **Martin Hachet**

13/05/1977

INRIA research scientist (CR1 – chargé de recherche 1ère classe)

Iparla project-team (INRIA – Université de Bordeaux)

Research topics : **3D User Interfaces**, Virtual Reality, HCI, Computer graphics

Email: Martin.Hachet@inria.fr

Phone: +33 5 40 00 69 20

Fax: +33 5 40 00 66 69

Web: <http://www.labri.fr/~hachet>

Address: LaBRI – Université Bordeaux 1, 351 cours de la Libération, 33405 Talence Cedex, France

Positions // Fonctions

Since Sept. 2005 : Research scientist – INRIA Bordeaux – Sud-Ouest

2003 – 2005: ATER (Temporary Assistant professor) – Université Bordeaux 1

Feb. – March 2004: Research scientist – Human-Interface Engineering Lab , Osaka University

April-Sept 2000: Engineer (Intern) – Cimpa-EADS

Education // Formation

2000 - 2003: PhD in Computer Science at Université Bordeaux 1, France

(defended on December 18th 2003)

Title : «*Interaction avec des environnements virtuels affichés au moyen d'interfaces de visualisation collective*»

Supervisor : Pr. Pascal Guitton, Université Bordeaux 1

2000: Master degree in Computer Science – Image and Sound (year 2), Univ. Bordeaux 1 – *Rank 1st*

1999: Master degree in Computer Science (year 1), Univ. Bordeaux 1, done at UQAM, Montréal (Canada)

Participation to the Scientific Community // Implication dans la communauté scientifique

Program Chair

- IEEE 3DUI 2010 - *Symposium on 3D User Interfaces*, Program co-chair (leader)
- ACM VRST 2008 - *Symposium on Virtual Reality Software and Technology*, Program co-chair (+local organization)

Other Chairing

- JVRC 2010 - *Joint Virtual Reality Conference (EGVE/EuroVR/VEC)*, Demonstration co-chair
- JVRC 2009 - *Joint Virtual Reality Conference (EGVE/ICAT/EuroVR)*, Poster co-chair

Program committees

- IEEE VR 2011
- ACM Siggraph Asia 2010
- Eurographics 2008, 2009
- IEEE 3DUI 2007, 2008, 2009, 2010 - *Symposium on 3D User Interfaces*
- ISVC 2009, 2010 - *International Symposium on Visual Computing*
- ACM VRST 2008 - *Symposium on Virtual Reality Software and Technology*
- VRIC 2008 - *Virtual Reality International Conference*
- IPT/EGVE 2007 – *Eurographics Symposium on Virtual Environments*

Reviewing : ACM Siggraph, ACM CHI, IEEE VR, ACM UIST, ACM I3D, CG&A, IHCS, JMUI, GI, IHM

Guest editor IEEE Transaction on Visualization and Computer Graphics (vol. 16, no. 1), Jan/Feb 2010

PhD Jury: F. Dècle (Sept. 2009), S. Knoedel (Dec. 2009), L. Auguerreche (June 2010), S. Hilaire (Nov. 2010).

Advising // Encadrements

PhD Students

Jérémy Laviolle:	Since 2010, with Christophe Schlick <i>"3DUI for computer Graphics"</i>
Aurélie Cohé:	Since 2009, with Pascal Guitton <i>"Touch-based 3DUI"</i>
Florent Berthaut:	Since 2007, with Myriam Desainte-Catherine <i>"Human, Image and Sound Interaction"</i>
Sebastian Knoedel :	2006-2009, with Pascal Guitton <i>"Beyond Desktop: Designing Novel User Interfaces to Enhance 3D Exploration"</i>
Fabrice Dècle:	2006-2009, with Pascal Guitton <i>"3D User Interfaces for Mobile Devices"</i>

Master Theses

Arash Kian :	2009, <i>"Opportunistic Music"</i>
Olivier Augerau :	2009, <i>"Design and Development of a Multi-touch system"</i>
Samuel Hurel :	2009, <i>"Multi-touch 3D Manipulation"</i>
Pierre Rouanet :	2008, <i>"Human-Robot Interaction"</i>
Florent Berthaut :	2007, <i>"Using VR for Music"</i>
Raphael Ducom :	2007, <i>"Skecth-based Navigation"</i>
Fabrice Dècle :	2006, <i>"3D Navigation on Mobiles devices "</i>

Engineers

Fabrice Dècle :	Since 2009, ANR Project InSTInCT, <i>Touch-based 3DUI</i>
Mariam Amyra:	2007-2009, ANR Project Part@ge, <i>Collaborative Interaction on Mobile Devices</i>
Cédric Kervegant :	2007 (6 months), Univ. Bordeaux 1 Internship, <i>Immersive Projection Wall</i>
Julien Hadim:	2003 (6 months) Pôle Eitica, <i>Drivers and API for the CAT</i>
Christophe Periet:	2003, (6 months) Univ. Bordeaux 1 Internship, <i>Demonstartion suite for the CAT</i>
Fernando Duga:	2002, (6 months) Univ. Bordeaux 2 Internship, <i>VR for an artistic performance</i>

Student projects, short internships

I have advised students since Sept. 2000 (graduate and undergraduate level)

Grants and Collaborations // Contrats et Collaborations

- InSTInCT : Project leader
2009-2012, ANR (National Research Agency)
Touch-based 3DUI for interaction with 3D contents
Partners: INRIA Lille (Alcove), Immersion, Cap Science
<http://anr-instinct.cap-sciences.net/>
- “Improving the VR experience”*:
Collaborator
2008-2010, JST/CNRS Grant
Improving the VR Experience
Partners: LSSIIT Strasbourg, INRIA Rennes (Bunraku), Tokyo University, Keio University, and Osaka University
- Part@ge: Scientific leader for the Iparla project-team
2007-2010, ANR (National Research Agency)
Collaborative Interaction
Partners: CEA-LIST, Clarte, CNRS, ESIA, France-Telecom, Haption, INRIA Lille (Alcove), INSA Rennes, Renault SAS, Sogitec, Thales, Virtools
<http://partage.ingenierium.com/>
- Dalia: Collaborator
2007-2010, ANR (National Research Agency)
Grids for heterogeneous interaction
Partners: INRIA Grenoble (Moais, perception), Université d’Orléans
<http://dalia.gforge.inria.fr/>
- Raxenv: Collaborator
2007-2009, ANR (National Research Agency)
Outdoor Augmented Reality
Partners: BRGM, Lyonnaise des eaux, Université d’Evry, Archividéo.
<http://raxenv.brgm.fr/>
- SOUL: Collaborator
2007-2009, National Competitivity Cluster AESE
Visualization and interaction for autonomous embedded systems
Partners: Thalès, BeTomorrow, Axyz
<http://www.aerospace-valley.com/en/>

Other activities // Autres activités

- Expert for ANR (National Research Agency), “programme blanc internationaux” - 2010
- Expert for the National Competitivity Cluster OPTITEC - 2009, 2010
- Elected member of INRIA Bordeaux Centre Committee – Member of the steering board (since 2008)
- Member of the INRIA Bordeaux Technological Development Committee (since 2008)

Teaching // Enseignement

Virtual Reality and Image Synthesis

keywords: input and output devices, stereoscopic visualization, software and toolkits, scene graph, 3D user interfaces, interaction techniques, immersion, human factors.

2008-2009 Engineering school, *Institut de cognitive* [7h]

2001-2007 Master 2 - *Université Bordeaux 1* [120h]
2001-2003 Master 2 - *Université Bordeaux 3* [26h]
[Total 153h]

General topics of computer science

The following teachings were given at *Université Bordeaux 1*

2004-2005 Networks, Master 1 [30h]
2004-2005 Image and sound, Licence 3 [26h]
2003-2004 Computer architecture, Licence 3 [35h]
2003-2004 Introduction to computer science, Licence 1 [35h]
2001-2003 C programming, Licence 2 [102h]
2001-2002 Fundamentals in computer science, Licence 1 [10h]
[Total 238h]

Technology transfer and softwares // Transfert technologique et logiciels

- The CAT [30] is available as a commercial product at Immersion [www.immersion.fr]
- Navidget library [<http://iparla.labri.fr/software/navidget/>] – more than 500 downloads

Communications and demos for the general public // Diffusion de l'information scientifique

I have been directly implied in demonstrations, in particular:

«Immersive Live-looping», Jeudis multimédias – LaBRI, May 2010
«Collaborative User Interfaces», Paris Air Show Le Bourget - Aerospace Valley, June 2009
« Immersive Music », Fête de la science - LaBRI, November 2009
« Navidget », European Research and Innovation Exhibition - Grand Palais Paris, November 2008
« The CAT », European Research and Innovation Exhibition - Parc des expositions Paris, June 2006
« The CAT », Siggraph Exhibition, Los Angeles, August 2005
« Tangimap », Innovation days - Biarritz 2005

I participate to the diffusion of scientific diffusion through dedicated websites and magazines: Interstices (article), Inedit (article), INRIA mediatheque (photos and videos).

Best paper awards // Prix de meilleurs papiers

- IEEE 3DUI 2008 - *Symposium on 3D User Interfaces* [15]
- ACM VRST 2003 - *Symposium on Virtual Reality Software and Technology* [30]

Publications

⇒PDFs and Videos are available on my webpage: <http://www.labri.fr/~hachet>

Books Editor

- [1] Martin Hachet, Kiyoshi Kiyokawa, Joseph LaViola, Proceedings of the 5th IEEE Symposium on 3D User Interfaces (3DUI 2010) IEEE - ISBN: 978-1-4244- 6844-7, 2010
- [2] Bernd Froehlich, Ernst Kruijff, Martin Hachet, Proceedings of the 15th ACM Symposium on Virtual Reality Software and Technology (VRST 2008) -ISBN:978-1-59593-951-7- 2008

Book Chapter

- [3] Sabine Coquillart, Philippe Fuchs, Jerome Grosjean, Martin Hachet, Le traité de la réalité virtuelle, Vol. Interfaçage, immersion et interaction en environnement virtuel - Les techniques d'interaction. Presses de l'Ecole des Mines, Volume 2, page 329--380 - March 2006

Articles in journals

- [4] Martin Hachet, Fabrice Dècle, Sebastian Knödel, Pascal Guitton. Navidget for 3D Interaction: Camera Positioning and Further Uses. *Int. J. Human–Computer Studies* – (vol 67, issue 3), p225-236 – March 2009
- [5] Martin Hachet, Joachim Pouderoux, Pascal Guitton. 3D Elastic Control for Mobile Devices. *IEEE Computer Graphics and Applications*, (vol. 28, no 4), p 58--62 - July/August 2008
- [6] Martin Hachet, Pascal Guitton, Patrick Reuter, Florence Tyndiuk. The CAT for Efficient 2D and 3D Interaction as an Alternative to Mouse Adaptations. *Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, vo. 23, no 3, p 728 - August 2004 – Reprise session of VRST 2003

Additional publications in journals

- [7] Martin Hachet and Ernst Kruijff, Guest Editor's Introduction: Special Section on the ACM Symposium on Virtual Reality Software and Technology, *IEEE Transactions on Visualization and Computer Graphics* (vol. 16, no. 1), p 2-3 - Jan/Feb 2010

International Conferences and Workshops

- [8] Berthaut, Florent; Desainte-Catherine, Myriam ;Hachet, Martin, «DRILE: an immersive environment for hierarchical live-looping », *Proceedings of New Interfaces for Musical Expression (NIME 2010)* - 2010
- [9] Berthaut, Florent; Hachet, Martin; Desainte-Catherine, Myriam, «Combining audiovisual mappings for 3D musical interaction», *Proceedings of International Computer Music Conference (ICMC 2010)*– 2010
- [10] Berthaut, Florent; Hachet, Martin; Desainte-Catherine, Myriam, « Piivert: Percussion Interaction for Immersive Virtual Environments », *Proceedings of Symposium on 3D User Interfaces (3DUI2010)*– 15—18 (4 pages) - 2010
- [11] Berthaut, Florent; Desainte-Catherine, Myriam; Hachet, Martin, «Interaction with the 3D reactive widgets for musical performance.», *Proceedings of Brazilian Symposium on Computer Music (SBCM09)*, (2009)
- [12] Decle, Fabrice; Hachet, Martin, «A Study of Direct Versus Planned 3D Camera Manipulation on Touch-Based Mobile Phones», *Proceedings of International Conference on Human-Computer Interaction with Mobile Devices and Services (Mobile HCI2009)*, 1--5 (2009)
- [13] Knoedel, Sebastian; Hachet, Martin, Guitton, Pascal , « Interactive Generation and Modification of Cutaway Illustrations for Polygonal Models », *Proceedings of International Symposium on Smart Graphics* , 140--151 (2009)
- [14] Decle, Fabrice; Hachet, Martin; Guitton, Pascal, «ScrutiCam : Camera Manipulation Technique for 3D Objects Inspection», *Proceedings of the IEEE Symposium on 3D User Interfaces (3DUI)* (4 pages) 19--22 (2009)
- [15] Hachet, Martin; Arash, Kian; Berthaut, Florent; Franco, Jean-Sébastien; Desainte-Catherine, Myriam, «Opportunistic Music», *Proceedings of JVRC 2009 (EGVE - ICAT - EuroVR)* 45—51 (2009)
- [16] Hachet, Martin; Decle, Fabrice; Knoedel, Sebastian; Guitton, Pascal, «Navidget for Easy 3D Camera Positioning from 2D Inputs», *Proceedings of the IEEE Symposium on 3D User Interfaces (3DUI)* 83--88 (2008) Best Paper Award
- [17] Hachet, Martin; Kulik, Alexander, «Elastic Control for Navigation Tasks on Pen-based Handheld Computers», *Proceedings of the IEEE Symposium on 3D User Interfaces (3DUI2008)* 91--96 (2008)
- [18] Knoedel, Sebastian; Hachet, Martin; Guitton, Pascal, «Navidget for Immersive Virtual Environments», *Proceedings of 15th Symposium on Virtual Reality Software and Technology (VRST 2008)*, 48--50 (2008)

- [19] Zendjebil, Imane; Ababsa, Fakhr-Eddine; Didier, Jean-Yves; Vairon, Jacques; Frauciel, Luc; Hachet, Martin; Guitton, Pascal; Delmont, Romuald, «Outdoor Augmented Reality: State of the Art and Issues», Proceedings of 10th ACM/IEEE Virtual Reality International Conference (VRIC 2008) 177--187 (2008)
- [20] Hachet, Martin; Pouderoux, Joachim; Tyndiuk, F.; Guitton, Pascal, «Jump and Refine for Rapid Pointing on Mobile Phones», Proceedings of CHI 2007 (4 pages) 167--170 (2007)
- [21] Knoedel, Sebastian; Hachet, Martin; Guitton, Pascal, «Enhancing the Mapping between Real and Virtual World on Mobile Devices through efficient Rendering», Proceedings of Mixed Reality User Interfaces 2007 (IEEE VR Workshop) (2007)
- [22] Knoedel, Sebastian; Pouderoux, Joachim; Hachet, Martin; Guitton, Pascal, «3D Panorama Service on Mobile Device for Hiking», Proceedings of CHI Workshop on Mobile Spatial Interaction 106--109 (2007)
- [23] De La Rivière, J.-B.; Ditlo, N.; Hachet, Martin, «Innovative 6DOF Devices for Desktop Configurations», Proceedings of Virtual Concept électronique (2006)
- [24] Hachet, Martin; Watanabe, R.; Kitamura, Y., «A Collaborative Interface for IllusionHole using a Control-Ring and Mice», Proceedings of IEEE Symposium on 3D User Interfaces (3DUI) -Technote-, + One page (Japanese) in Proc. of the 2006 IEICE General Conference 66-68 (2006)
- [25] Hachet, Martin; Declé, Fabrice; Guitton, Pascal, «Z-Goto for Efficient Navigation in 3D Environments from Discrete Inputs», Proceedings of Virtual Reality Software and Technology (VRST) (4 pages)236--239 (2006)
- [26] Hachet, Martin; Pouderoux, Joachim; Guitton, Pascal; Gonzato, Jean-Christophe, «TangiMap - A Tangible Interface for Visualization of Large Documents on Handheld Computers», Proceedings of Graphics Interface (GI 2005) 9 - 15 (2005)
- [27] Hachet, Martin; Pouderoux, Joachim; Guitton, Pascal, «A Camera-Based Interface for Interaction with Mobile Handheld Computers», Proceedings of Symposium on Interactive 3D Graphics and Games (I3D2005), 65 - 72 (2005)
- [28] Hachet, Martin; Kitamura, Y., «3D Interaction With and From Handheld Computers», Proceedings of IEEE VR 2005 Workshop: New Directions in 3D User Interfaces électronique (2005)
- [29] Hachet, Martin; Guitton, Pascal, «The CAT - When Mice are not Enough», Proceedings of IEEE VR 2004 Workshop: Beyond Glove and Wand Based Interaction 66--69 (2004)
- [30] Hachet, Martin; Reuter, Patrick; Guitton, Pascal, «Camera Viewpoint Control with the { Interaction Table} », Proceedings of Virtual Reality International Conference (VRIC 2003) 41--47 (2003)
- [31] Hachet, Martin; Guitton, Pascal; Reuter, Patrick; Tyndiuk, F., «The CAT for efficient { 2D} and { 3D} interaction as an alternative to mouse adaptations», Proceedings of ACM Virtual Reality Software and Technology (VRST'03) 205-212 (2003). Best Paper Award
- [32] Hachet, Martin; Guitton, Pascal, «Using Virtual Reality for New Clowns», Proceedings of International Conference on Virtual Storytelling 211--219 (2003)
- [33] Hachet, Martin; De La Rivière, J.-B.; Guitton, Pascal, «Interaction in Large-Display VR Environments», Proceedings of Virtual Concept électronique (2003)
- [34] Hachet, Martin; Guitton, Pascal, «The Interaction Table: a New Input Device Designed for Interaction in Immersive Large Display Environments», Proceedings of Eurographics Workshop on Virtual Environments (EGVE 2002), Barcelone 189--196 (2002)

National Conferences (with proceedings)

- [35] Berthaut, Florent; Desainte-Catherine, Myriam; Hachet, Martin, «Widgets Réactifs 3D», Actes des Journées d'informatique musicale (2009)

- [36] Berthaut, Florent; Desainte-Catherine, M.; Hachet, Martin, «Interaction 3D pour la musique», Actes des 2emes Journées de l'AFRV 75 -- 81 (2007)
- [37] Knoedel, Sebastian; Hachet, Martin; Guitton, Pascal, «Sketch-based Route Planning with Mobile Devices in immersive Virtual Environments», Actes des 2èmes journées de l'AFRV (2007)
- [38] Declé, Fabrice; Hachet, Martin; Guitton, Pascal, «Z-Goto: pour une Navigation Efficace dans des Environnements 3D sur Terminaux Mobiles», Actes des 1eres Journées de l'AFRV 75--82 (2006)

Invited talks

“3D Multitouch interaction”, V-City Workshop (EU project – FP7), Bordeaux, June 2010

“Virtual Reality for Music”, Bunraku Seminar, Rennes, June 2010

“3D User Interfaces - from immersive environments to mobile devices”, 2nd Sino-French Workshop, Huangshan (China), August 2009

“3D User Interfaces - from immersive environments to mobile devices”, REVES seminar, Sophia-Antipolis, March 2008

“Mobile 3D User Interfaces.” Bauhaus – Universität Weimar (Allemagne) Virtual Reality Group – Seminar, December 2006.

Interaction avec des mondes 3D : quand la souris ne suffit plus. *Journée « La simulation au service de l'innovation »*. ESTIA, Biarritz, avril 2005.

3D interaction from large screens to small devices. *Seminar*. Human interfaces engineering lab, Osaka (Japon), janvier 2005.

3D interaction from large screens to small devices. *Seminar*, Cybermedia center, Osaka (Japon) février 2005.

The CAT – Control Action Table. *Séminaire*. Imager – UBC, Vancouver (Canada), juillet 2004.

Introduction

The digital revolution has changed usages in any domain of our Northern societies¹. In particular, interactive 3D graphics play an important role in this evolution. This powerful medium allows users to better understand, learn, conceive, or exchange with each-other. For example in aviation industry, the whole design of new planes entirely relies on CAD software solutions. The widespread boom surrounding interactive 3D applications is strongly linked to the impressive advances of computer graphics technologies that have occurred these past few years, both in the scope of hardware and software developments. Compare to the amount of work that has been done to address the challenges coming from these technological concerns, smaller amount of work has focused on interfaces allowing one to interact well with 3D content. However, these interaction issues are crucial in interactive 3D graphics, and the success of any 3D application is strongly linked to the relevance of its user interfaces (UI). In this document, I will focus on the challenges of 3D interaction in different technological contexts, and I will present the works I have been implied in, from my PhD thesis in 2000-2003 to the most recent work I currently focus on. In particular, I will discuss 3D UIs for three different contexts: mobile interaction, (multi)touch-based interaction, and immersive interaction.

Research area The question of how users can communicate with interactive systems has gone with the story of computer science. Human-Computer Interaction (HCI) is the science that addresses such a question. Since Yvan Sutherland's pioneering work on graphic interaction (PhD thesis at MIT Lincoln Laboratory, 1963), the general HCI community has changed the way we interact with digital data, mainly in 2D spatial contexts.

At the same time, the computer graphics (CG) community has brought interactive 3D graphics to a mature state, where complex 3D data can be displayed in real-time on various platforms, from powerful workstations to small mobile devices. To achieve this goal, the whole graphic pipeline (acquisition, modeling, rendering) has been addressed. On the other hand, interaction issues have often been set aside. Since the eighties where 3D control widgets were introduced [Bie87], desktop user interfaces dedicated to interaction with 3D content have few evolved.

For the Virtual Reality (VR) community, the quest has been to immerse users in multi-sensory synthetic worlds, since the invention of the Sensorama Machine by Morton Heilig in 1957. The conjunction of numerous domains such as CG, HCI, cognition and psychology, haptics, systems and display technologies has contributed to this general quest. An important objective of VR is to reach a good immersion by insuring efficient interaction processes between the user and the virtual environments. Consequently, numerous input devices and interaction techniques dedicated to 3D interaction have been developed to achieve this goal, in an immersive context.

Recently, a new research community centered on 3D User Interfaces (3DUI) has emerged from the three above-mentioned communities. In 2004, the book "3D User Interfaces - Theory and Practice"

¹According to the International Telecommunication Union (ITU), less than 4% of african people have access to internet.

[BKLP04] was published to aggregate the knowledge in this area. At the same time, two workshops dedicated to the questions that are linked to interaction with 3D data were organized by Doug Bowman, Bernd Froehlich and Yoshifumi Kitamura during the annual IEEE VR conferences (2004, 2005). The success of these workshops led to the creation of the 3DUI annual symposium, which is now well known and well established as a major event.

A 3D user interface is an interface that involves 3D interaction. According to Bowman et al. *"3D Interaction is a Human-Computer Interaction in which the user's tasks are performed directly in a 3D spatial context. Interactive systems that display 3D graphics do not necessarily involve 3D interaction; for example, if a user tours a model of a building on her desktop computer by choosing viewpoints from a traditional menu, no 3D interaction has taken place. On the other hand, if the user clicks on a target object to navigate to that object, the 2D mouse input has been directly translated into a 3D location, and thus 3D interaction has occurred."*

3DUI is at the frontier between HCI, CG, and VR. It is a sub-research area of HCI where the spatial context is 3D. It lies on CG where the focus is the user. Finally, 3DUI share some goals with VR, but it is not restricted to immersive contexts. Desktop configurations, mobile devices, tabletop systems, VR and AR configurations are examples of technological contexts where 3D UIs need to be designed to help users interacting with 3D graphics. Figure 1 illustrates this positioning.

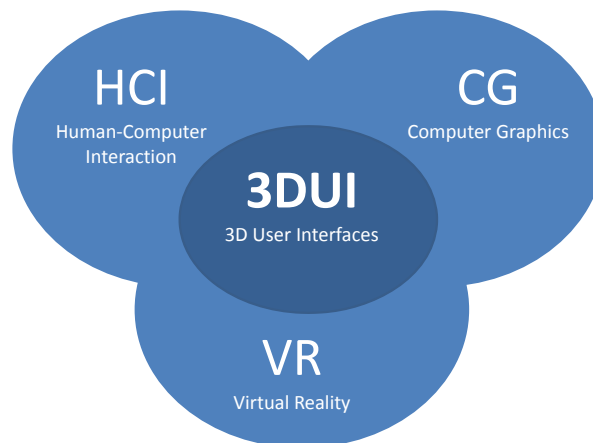


Fig. 1: 3DUI positioning.

Similarly to any UI, a 3D UI operates in two directions. Users input some actions to modify the 3D environments or to change the state of the application (e.g. modify the viewpoint). In the other direction, users are informed of the changes by way of appropriate feedbacks. To insure this bi-directional communication, both hardware and software components are implied. Standard hardware interfaces may be used to interact with 3D graphics. For example, a regular mouse associated with efficient interaction techniques may be well suited to complete some interaction tasks with 3D data displayed on standard LCD screens. For other 3D interaction tasks, the standard hardware interfaces are not well suited anymore, and new input devices have been designed to better fit the structure of 3D space. In particular, multi degrees-of-freedom (DOF) input devices allow one to control several 3D

parameters at the same time (eg. 3 translations and 3 rotations). Similarly, dedicated output devices such as stereoscopic and haptic devices, can be valuable to 3D interaction.

My personal background and evolutions I started my PhD in 2000, in Bordeaux, under the supervision of Pascal Guitton. At this time, the LaBRI was being equipped with a SGI Reality Center called Hemicyclia. This immersive setup was composed of a large curved screen (10x3m), three CRT Barco projectors, and a SGI machine. My initial PhD subject was to explore the benefit of such an immersive setup for information visualization. In this context, I started working on new metaphors to enhance the understanding of abstract data displayed on geographic layers. One of the directions I explored was the use of 3D tubes, in which bubbles with different shapes, appearances, and movements were linked to some parameters (eg. nb of employees in a shop). One of the main problems I faced during these experiments was the difficulty for users to interact with the 3D environments. Indeed, it is very difficult for users to understand well the data they are dealing with if they are not able to interact with them in a simple and efficient way. For example, comparing the relative sizes of two objects in 3D requires extensive modifications of the camera viewpoint, which may be hard to perform. Consequently, I redirected my research toward interaction, which I have considered as the main challenge to address before going farther with any 3D interactive applications. At this time, this statement has been strengthened by some discussions I had with companies during workshops of PerfRV, the national research network dedicated to Virtual Reality. Indeed, the companies tended to express satisfaction concerning the complexity of the 3D models they were using as well as the quality of the images they were able to display, but they expressed a true difficulty as soon as users had to interact with the data. The idea of the CAT, a 6 DOF input device designed to enhance interaction with 3D content displayed on large screens came at this time. This hardware interface, which led to several publications and technological transfers, will be described in more depth in the third part of this document.

In 2002, INRIA decided to create a new research center in Bordeaux. We took benefit from this opportunity and we created a joint project-team (INRIA - Université de Bordeaux/LaBRI) called Iparla. The initial target of Iparla was to visualize and to interact with complex data on mobile devices. In this context, I reoriented my research towards mobile devices. My goal has been to move and adapt 3DUI from desktop to mobile setups in order to favor the creation of interactive 3D applications on mobile devices. Indeed, insuring real-time rendering of 3D objects on mobile devices is not enough to bring 3D applications to mobility. New 3D UIs need to be conceived, and this challenge has been the focus of my research activities in the Iparla project-team. This objective allowed me to explore new directions that will be presented in the first part of this document. In 2005, I have been recruited at INRIA as a permanent researcher.

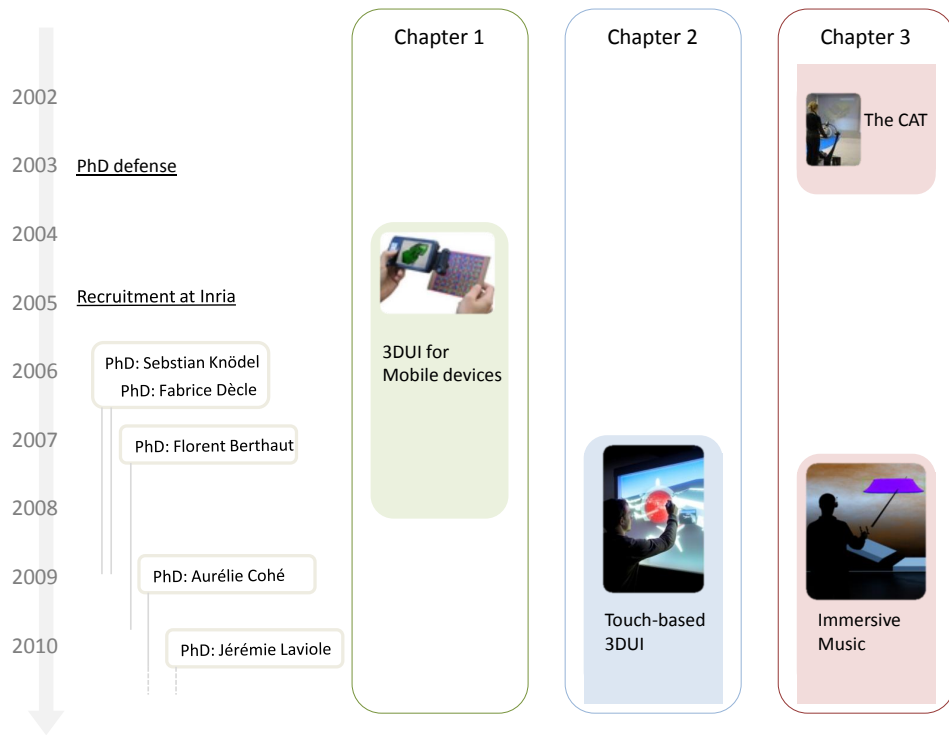
In 2006, we recruited two PhD candidates in the scope of 3DUI. This allowed us to create an "interaction group" in the Iparla project-team. The first one, Fabrice Dècle, has focused his research on 3D UIs for mobile devices. The second one, Sebastian Knoedel, has explored some new directions to reduce the gap between mobile and immersive contexts, and to have these two separate technological contexts working together. In 2007, Mariam Amira joined the group as an engineer, and we have contributed to Part@ge, a national ANR research project, where the goal was to enhance collaborative interaction for mobile users.

Beyond 3DUI and computer graphics dedicated to mobile devices, we have changed the objectives of the Iparla project-team -and in particular the interaction group- towards a more general mobile context. The goal is not to focus only on mobile devices anymore, but more to think about new approaches

aiming at favoring the mobility of users, from mobile devices to immersive virtual environments, and from standard desktop configurations to collaborative tabletop setups. Indeed, the standard workstation paradigm where users interact by way of keyboards and mice is evolving. Today, new systems and interfaces allow users to interact with digital content in different locations and at different times. In particular, the world is witnessing a widespread boom surrounding interactive touch-based surface technologies, noticeably with the extremely rapid market penetration of multi-touch surface technologies, which have lately been spreading ubiquitously from small devices and smartphones to large, multimedia touch-screens. In this context, we have explored new approaches aiming at tackling the challenges of 3DUI for touch-based interaction. Our first results will be described in the second part of this document. We have just started a project focusing on this topic, InSTInCT, a national ANR project, which aims at bringing 3D interfaces from its current office desktop use to new, broader usage and contexts. Aurélie Cohé has been recruited as a PhD student for this project, and Fabrice Dècle will participate as an engineer.

At the same time, since 2007, I have explored new directions in the scope of music with Florent Berthaut, a PhD student I co-advise with Myriam Desainte-Catherine. The purpose of this work is to use and to contribute to the evolution of immersive technologies in order to provide rich interaction between musicians, audiences, images and sounds. This work is very motivating for me as it targets artistic creation. It is also an inspiring context where we explore new concepts. The development of our virtual instrument led to interesting results in the scope of audio-visual mappings and hierarchical structures, design of new input devices, and experimentations with various display configurations. This work will be described in more depth in the third part of this document.

Visual Overview



1.1 Introduction

Mobility is a major (r)evolution for current interactive systems. Mobile devices that were dedicated to phone calls only few years ago, tend to be used as personal computers, now. In addition to short texts, sounds, images and videos, mobile users may now benefit from interactive 3D graphics displayed on their mobile devices. This evolution has been supported by the impressive recent advances in mobile technologies . New usages have appeared from this technological evolution. For example, mobile users may benefit from 3D visualization of their surrounding environments for assisted navigation. Figure 1.1 illustrates such an application we have developed for hiking purposes [22]¹. Another example is the use of interactive 3D graphics for cultural heritage where users visualize, on-site, 3D reconstructions of buildings that disappeared. Many other examples could be cited in the scope of maintenance, entertainment, sports, and so on.

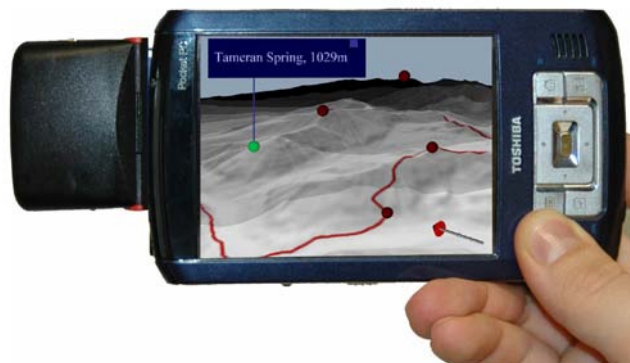


Fig. 1.1: Example of 3D mobile application: a hiking assistant [22].

Despite the potential benefit of such mobile 3D applications, few concrete examples have reach a general public success, except video-games on dedicated mobile consoles. We think that one of the reasons that limits this general acceptance is the difficulty for users to interact well with 3D data displayed on mobile devices. Few user interfaces have been specifically designed to favor mobile 3D interaction. Consequently, the level of interactivity with such settings is generally much reduced.

Most of 3D UIs have been designed for desktop configurations or VR setups. Consequently, they do not fit well with the constraints of mobile devices (eg. mobile ergonomic issues and hardware limitations, see the PhD thesis of Fabrice Dècle for details [D09]). General HCI has evolved towards mobile-HCI. In particular, the research community has been very active to adapt 2D UIs to the mobile

¹References in italic refer to my personal publications - see curriculum vitae at the end of this document

input and output spaces (e.g. Shift technique [VB07] and MicroRolls [AEY09]). The mobile input space has also been increased (e.g. LuscidTouch [WFB⁺07]). I am convinced that 3DUI should follow a similar evolution, from its current state to mobile-3DUI. This evolution is required to enhance 3D interaction on mobile device and, consequently, to favor the development of new 3D applications for mobile users.

In the following sections, I present some examples of mobile 3D UIs we have developed since 2003. These examples follow the evolution of the input strategies on mobile devices. In a first step, I present two results based on keystroke interaction. Then I focus on tactile interaction. Finally, I present two prototypes that increase the standard mobile device input space for 3D interaction.

1.2 Interaction from keystrokes

Until a recent time, mobile devices were mainly based on discrete inputs for interaction with the system. Directional keys and small discrete thumb-joysticks were the standard physical interfaces of the mobile devices. Examples are given in Figure 1.2. These input devices, controlled by the thumb motions are generally used to "jump" from an item to another. This jump-based navigation is well suited for navigation in discrete structure such as 1D lists (eg. list of contact) or 2D lists (eg. arrays of icons). This paradigm is widely accepted by the general public, and younger users who are frequent mobile phones users have even developed new anatomical skills for interaction from their thumb.



Fig. 1.2: Example of key-based mobile devices.

The problem comes when users need to interact with continuous structure such as images or 3D environments. In these cases, the standard approaches for the completion of interaction tasks may become inefficient when completed from keystrokes. In particular, moving a cursor over the screen to reach each part of the displayed content may be time consuming. We interested in this problem, and proposed two interaction techniques to better adapt key-based input to interaction with continuous content on mobile phones.

1.2.1 Jump and Refine

The first technique called "Jump and refine" aims at improving performance of users in pointing tasks. This technique is not dedicated to one or few specific applications. It rather aims at operating as a universal technique on key-based mobile phones. The main idea is to reduce the number of keystrokes by using two (or more) successive levels. In the first level (jump), a grid is displayed on the screen as illustrated in Figure 1. The cursor is positioned at the center cell. From directional inputs on the phone, the cursor is moved by successive jumps from cells to cells. This allows fast cursor movements. Then, a pressure on the "validate" key leads to the second level (refine) where the only current cell is

displayed as a visual feedback. In this level, the cursor can precisely be moved for accurate positioning if necessary, as illustrated in Figure 1.3.



Fig. 1.3: The two-levels "Jump and Refine" selection technique.

We interested in the optimal grid size for a given screen resolution and at a given precision. For example, with a horizontal resolution of 176 pixels, which was a standard mobile phone resolution, and with a precision of 3 pixels, the number of keystrokes required to go from the center to the border of the screen is 29. Now, if we use our jump and refine approach with a grid composed of 7 columns, this number decreases to 7. In [20], we show that this number of keystrokes is given by

$$Num_{max} = \left\lfloor \frac{n}{2} \right\rfloor + \left\lceil \frac{resolution}{2.n} \right\rceil \quad (1.1)$$

with n the number of grid columns (resp. lines) and *resolution* the horizontal (resp. vertical) resolution of the screen. The optimal number of lines (resp. columns) is reached when the derivative of (1.1) is zero, that is

$$n = \sqrt{\frac{resolution}{precision}} \quad (1.2)$$

In our example, this formula says that 7 is the optimal horizontal grid size to minimize the number of keystrokes. We conducted an experiment to verify that the number of keystrokes was correlated to the users' completion time for a pointing task on a mobile phone operated by directional keys. We found that the performance of the users followed the theoretical number of required keystrokes. Moreover, the suggestive rating of the grid sizes by the subjects was concordant with the theoretical prediction and experimental results. Additional details can be found in [20].

Applications where pointing tasks are required are not well suited to be used with key-based mobile devices. With Jump and Refine, we showed how to reduce the gap between such applications and the input devices based on discrete events. Beyond mobile devices, the Jump and Refine approach can be used in various domains. For example, the majority of remote controls are equipped with directional keys. Jump and Refine could thus be used for selection tasks with interactive television.

1.2.2 Z-Goto

Similarly to Jump and Refine, Z-Goto is a technique we developed to improve user performance with key-based mobile devices. Whereas Jump and Refine addresses the general problem of pointing independently of the application, Z-Goto has been designed specifically to improve navigation in 3D environments. 3D navigation from keystrokes is difficult. For example, a continuous approach where the user controls the speed and the direction of the camera movements from discreet input is very limited as soon as the environment becomes large. With Z-Goto, we explored a *Go to* approach where the user indicates the location he or she wants to fly to.

Compared to a standard Point-Of-Interest technique [MCR90] where the user selects the end point of the trajectory by picking the projection of the target on the screen plane, Z-Goto operates in depth, as follow. At any time, the depth of the current view is divided up in a few numbers of sections. The current section is highlighted by a colored strip, as illustrated in 1.4. Hence, with few keystrokes, users can select target depths from nearest to farthest locations. This DOF relates to the distance where the users want to fly to. Once the depth range is selected, a target point is highlighted (the blue point on Figure 1.4). Its size depends on its distance from the viewpoint, in order to improve depth perception. This target point can then be moved from left to right to select the target destination, directly in the 3D space. Additional details can be found in [25].

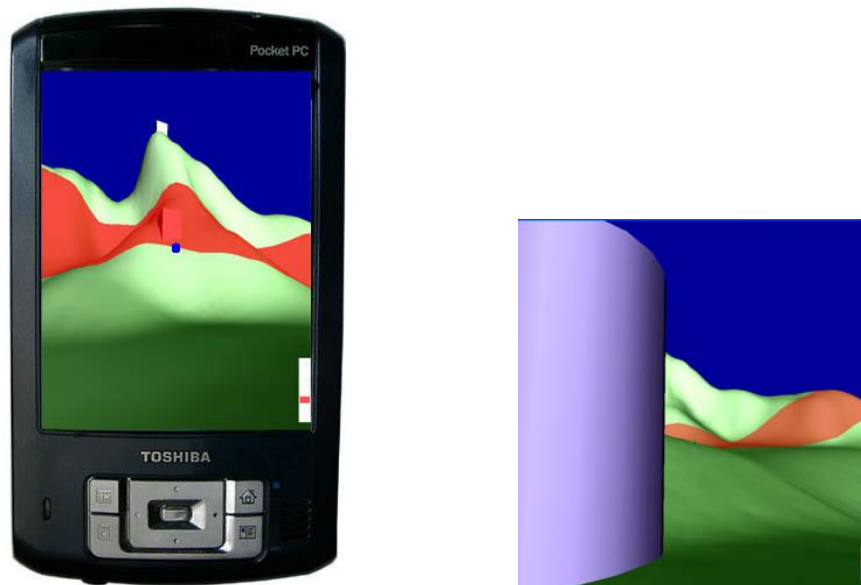


Fig. 1.4: Z-Goto for 3D navigation from keystrokes

Z-Goto has been designed to be used with key-based mobile devices. Our main motivation was to limit the number of required keystrokes by taking into account the underlying 3D environment structure. For example, in Figure 1.4-right, the left part of the image is a first-plane object for which a unique targeting should be necessary. The upper-right part fits to the sky for which no targeting should be possible. At the end, the only lower-right part of the screen has to be reached for navigation. With Z-Goto the target point is moved in this interesting area only, which optimizes the number of required keystrokes. Moreover, by sliding the current section along the z axis, users may improve perception of the depth, which may help them for the construction of cognitive maps. Sliding colored strips inside the scene favors the perception of the contours that can be difficultly perceived on small screens. For example, in the case of 3D terrains, the relief of the ground can be better felt.

We conducted a preliminary experiment to assess the usability of Z-Goto (see [25] for details). It showed that Z-Goto can be a good alternative to a standard *Go to* technique where a cursor is moved on the screen plane by way of successive keystrokes. In particular, the subjects reported that Z-Goto was faster, and it allowed them to better perceive the 3D environment, which contribute to our general quest of improving immersion on small handheld devices. Beyond mobile devices, Z-Goto can benefit to desktop users, too. Navigation can be done by way of keyboard strokes with the non-dominant hand, while the dominant hand controls the mouse for other tasks (eg. object selection).

1.3 Interaction from the mobile device touch-screens

During the past few years, keystroke interaction on mobile devices has evolved towards tactile interaction. The first devices taking benefit from this tactile approach were the PDAs, where styluses are used for efficient pointing and handwriting. Later, many mobile phones have adopted this new input modality, but the stylus techniques have tended to be replaced by UI directly operated from the users' fingers. Tactile mobile devices has changed the general mobile HCI, and this new input modality has been widely accepted by the users. On the other hand, few works have explored how this tactile paradigm adapts to 3D interaction tasks. In this section, I present two works that explore tactile input for 3D interaction on mobile phones. The first one is the result of a collaboration with Alexander Kulik, from Bauhaus Universitat - Weimar. It focuses on navigation for PDAs operated with styluses. The second one is dedicated to the observation of 3D objects on mobile phones where the users interact directly from their thumbs.

1.3.1 Improving stylus interaction with elastic feedback

The standard usage of PDAs is oriented towards 2D content, where users access and manipulate the data by directly "touching" them. Due to the congruency of the tactile input space and the graphical output space, the isotonic input that is based on the motion of a pen or fingers directly on the screen is very intuitive and effective. Interaction with mobile device applications is therefore mostly direct and position controlled. This one-to-one mapping is broadly accepted as working well for hypertext and menu interaction or scrolling through text and image documents.

On the other hand, 3D applications may require interaction tasks where this one-to-one mapping is not well suited anymore. In particular, navigation in large 3D environments requires potentially infinite viewpoint motion. For tasks like these, position-controlled techniques as currently provided with pen-based interaction are not very appropriate since they frequently require disturbing and irksome re-clutching. Rate control techniques seem to be more adequate. Zhai [Zha95] demonstrated the superiority of elastic devices for rate control techniques regarding a 3D docking task. Following his findings, the isotonic resistance characteristic of the pen does not seem to be well suited for the rate control required in 3D navigation tasks. Consequently, we developed a simple concept to adapt the input characteristics of a handheld computer to benefit from an elastic feedback for the control of viewpoint trajectories (see Figure 1.5).

We evaluated the differences between isotonic and elastic controls for 3D trajectory tasks on mobile devices. Much like a slalom competition, our evaluation task consisted of passing gates located at different depths as fast as possible. We were interested in differences regarding the completion times and the efficiency of the user trajectories assessed as the number of collisions that take place during



Fig. 1.5: A simple rubber band adds elastic feedback to pen input on a PDA.

the task. We also looked at the user preference in both conditions. The details of this experiment are described in [17].

We found that elastic control was significantly faster (28%) than isotonic control. We also found large differences between the two interfaces for the efficiency of the trajectories. These results, illustrated in Figure 1.6 demonstrate an important gain of control with elastic feedback given to the user. This has been strengthened by the subjective evaluation where the subjects reported that elastic feedback helped them for completing the task effectively. Under the isotonic condition, users adapt their movements from the visual feedback they obtain from the application. This visual feedback is the only output that guides them. On the other hand, elastic feedback gives users a stronger perception of their actions. Consequently, they are able to interact in a more efficient way.

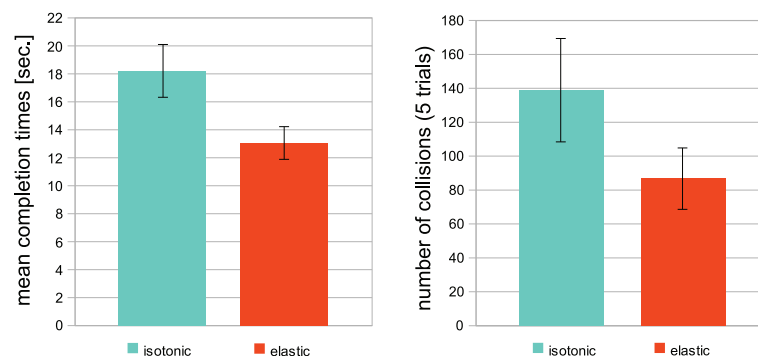


Fig. 1.6: Mean task completion times and average number of collisions per subject.

From these findings, we built prototypes taking benefit from elastic feedback for rate-controlled tasks such as 3D navigation, while maintaining the possibility for the user to interact in a more traditional way (eg. picking a 3D object). These prototypes are illustrated in figure 1.7.



(a) Elastic pen insertion with small movement deviation. (b) Elastic feedback system made of elastic fabric to be used with pen or thumb. (c) Semi-elastic pen insertion, providing constrained isotonic input for one degree of freedom, while offering elastic feedback to the other. (d) Semi-elastic pen insertion as in (c), but providing constraints to the elastic degree of freedom in relation to induced input with isotonic movement.

Fig. 1.7: Examples of elastic control insertion on PDA.

1.3.2 Sketchball

Many current mobile phones are equipped with tactile screens, where users interact directly with their fingers on the screen, without stylus. The iPhone is a famous example of such devices. The standard UI that have been developed for this input modality are based on tapping and sliding techniques. This allows one to select items by clicking directly on them (eg. an application icon), or to scroll long lists of items by flicking them (eg. list of contacts). Flicking is one example of mobile adaptation where the standard scroll bars of desktop HCI have been replaced by gestures on mobile devices.

We believe that a similar adaptation should take place for 3DUI. For example, the virtual trackball metaphor is a well-known technique that has become a standard for observation of 3D objects in a desktop context (ie. operated by a mouse). On mobile devices there is no evidence that this technique remains efficient. Some inherent mobile constraints may affect the performance. In particular, the fact that the thumb of the user occludes a large part of the screen may be a problem when observing 3D objects, as illustrated on the left side of the Figure 1.8. Moreover, the anatomical constraints as well as the precision issues make the control of the virtual trackball harder with the thumb than with a mouse. Consequently, we have explored a new direction for the control of a trackball, where users manipulate objects by sketching horizontal and vertical strokes (see Figure 1.9, right).

In our implementation, horizontal strokes result in rotations of the virtual camera around the vertical axis while vertical strokes result in rotations around the horizontal axis (see Figure 1.9). After the user releases his or her thumb from the screen, the camera is smoothly rotated according to the inputted gesture. Hence, the user is able to move the camera around the 3D model by drawing successive strokes. The rotation angle between two successive views is set according to the required precision. Our experience with the technique has shown that a 45 degrees rotation angle is a good compromise between speed and precision. It allows good visualization of the 3D models while limiting the number of required strokes.

The sketch-based trackball we propose, call Sketchball, induces a discretization of the possible views. Consequently, it is not well suited for precise 3D orientation tasks. Our approach follows the sketching philosophy, where coarse results are obtained by way of simple and fast commands. In the context of interactive 3D applications on mobile devices, we can assume that accurate positioning is rarely required. Coarse manipulation approaches can be better suited as soon as they allow the user

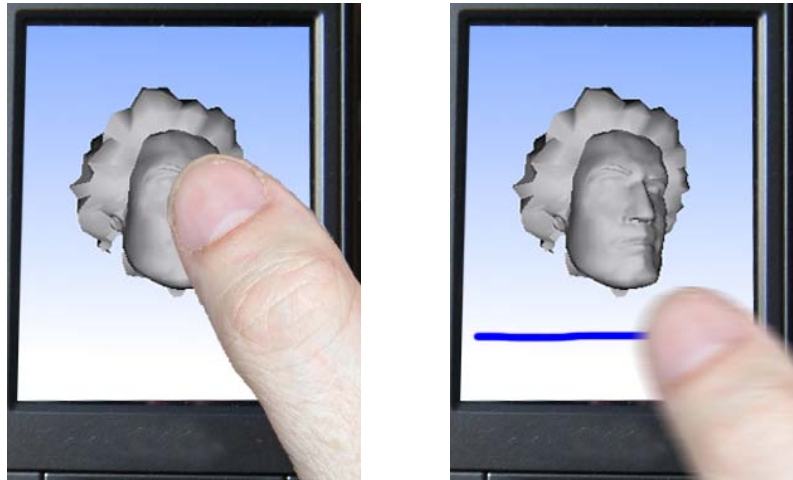


Fig. 1.8: A standard trackball on a mobile device induces occlusions of the screen (left). A stroke-based adapted trackball favors the visualization of the 3D model (right).

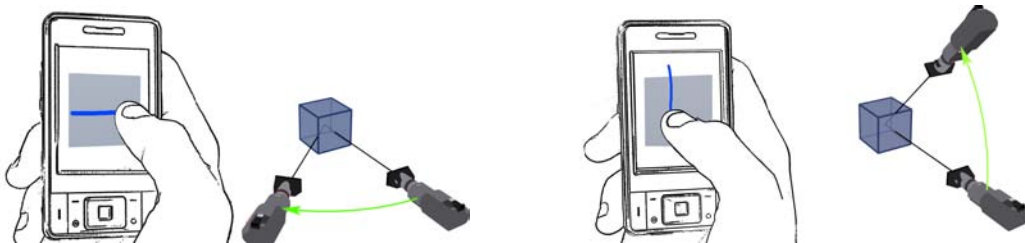


Fig. 1.9: Horizontal movements produce rotation around the “up vector” of the camera (top). Vertical movements produce rotation around the “Right vector” of the camera (bottom).

to understand well the 3D structure of the objects. To better understand the influence of the control (direct vs. sketched) on the user performance for an observation task, we conducted a preliminary user study. The task for this experiment consists in counting the number of targets drawn on the faces of a 3D model. We asked the subjects to perform the task twice, once with the standard trackball and once with Sketchball. The details of this experiment can be found in [12].

The results showed that the subjects performed the task faster with the direct-controlled trackball than with its sketch-based counterpart. This can be explained by the time needed to run the animation after the users’ strokes. On the other hand, this study showed that Sketchball helped the subjects to better maintain a spatial orientation. The subjects significantly reported that they felt lost with the standard trackball, while Sketchball helped them in the cognitive structuring of their movements. For example, by inputting two successive strokes in the same direction, the users know they will obtain a 90 degrees rotation, and they can come back to the original orientation by stroking twice in the opposite direction. Further investigations should be conducted to better understand the impact of the control on user performance. This work, even preliminary, showed us that UI based on gestures could be interesting alternatives to desktop 3DUI for interaction with 3D data displayed on mobile devices.

1.4 New inputs for mobile devices

In the previous sections, I presented some 3D UIs that were designed to be operated from the standard inputs of the mobile devices, ie. the keys and the tactile inputs. These UIs are very generic and,

consequently, they could be used directly on the current mobile devices. Beyond the standard mobile inputs, we explored the benefit of video-based interaction for mobile 3DUI. Indeed, the wide majority of the existing mobile devices are equipped with embedded cameras. Consequently, we tried to take benefit from this new modality to better adapt the mobile input space to the structure of 3D interaction tasks. In particular, we designed two multi-DOF interfaces based on video tracking. The first one called Tangimap explores bi-manual interaction. The second one is a 3 DOFs elastic controller.

1.4.1 Tangimap for bi-manual interaction

Our main objective with Tangimap was to explore proprioception² for the simultaneous control of 3 DOFs. To achieve this goal, we designed a bi-manual interface where users hold the mobile device with their non-dominant hand while moving a target with their dominant hand, as illustrated in Figure 1.10. With this configuration, they do not occlude the screen, which enhances the visualization of the displayed data. Moreover, they do not need to tilt the device to interact. Consequently, they can choose their optimal visualization angle to the screen in regard to the lighting conditions.



Fig. 1.10: Tangimap. By moving a target with his dominant hand, the user controls 3 DOFs for interaction with large documents or 3D environments.

In our implementation, the movements of the dominant hand from the frame of reference (ie. the non-dominant hand), are tracked thanks to the camera of the mobile device associated to a simple vision algorithm. The target, is a 12x12 cm wide square paperboard divided into an array of 8x8 cells. These cells are color-codes composed of 3x2 code-units. Each cell codes its position in the target using a binary representation where blue is assigned to 0 and green to 1, red being used as the background color. The three top code-units of a cell relate to the column number while the three bottom ones relate to the row number. By analyzing only a few pixels in each frame of the video stream captured by the camera, we are able to compute the x and y shift of the target from the referent position. The z coordinate is inferred from the cell width in pixel unit. A simple filter based on the previous records

²Proprioception is the unconscious perception of movement and spatial orientation arising from stimuli within the body itself (Source: The American Heritage Science Dictionary)

is used to avoid the jittering coming from the hand of the users. Additional details can be found in [27]. In our approach, we did not seek for an accurate and robust video tracking methodology. We rather developed a technique that computes user movements in a very fast way. Hence, the CPU is saved for end-user applications. The grid configuration allows us to track the target within a relatively large area, in particular when the user movements occur close to the camera.

With Tangimap, users are able to control 3 DOFs at the same time. This input structure is particularly well suited for 3D interaction tasks, as well as for interactive visualization of large documents (eg. maps and pictures). For manipulation of 3D objects, translation and rotation modes are separated. Two buttons of the mobile device operated from the users' non-dominant thumb allow users to switch between these two modes. Translations are directly mapped to the target movements. The mapping for the rotations is illustrated in Figure 1.11. For 3D navigation, the distance of the target from the camera allows one to control the speed of the movements in the 3D space, the 2 other DOFs being used to modify the yaw and pitch angles of the virtual camera. Hence, by moving the target behind the mobile device, users navigate in a 3D space with smooth and continuous movements. The semantic link between the target movements and the resulting actions in 3D spaces is relevant. This allows the cognitive immersion of users to be improved for 3D interaction tasks on mobile devices.

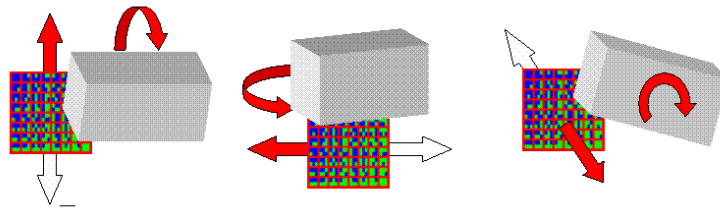


Fig. 1.11: Rotating 3D objects with Tangimap.

We assessed Tangimap for a large document visualization task. The task consisted in finding targets on a map. We asked 16 subjects to perform the task with Tangimap, and with a standard stylus approach. Details are given in [26]. We found that Tangimap was significantly faster. The user preference was widely in favor of Tangimap, too. By holding the tangible map, the users benefit from the proprioceptive feedback in addition to the visual feedback. For example when a target appears on the top-left corner, the subjects directly move the Tangimap target to a corresponding location. Then, the target can be found quickly. On the other hand, stylus interaction appears less efficient as the users have to move the map without knowing exactly where to go. The only feedback they have is the one provided by visual cues. At the time we conducted this experiment, no multi-touch mobile device was available. Now, it would be interesting to evaluate a multi-touch approach for a similar task.

Proprioception and multi-DOF interaction has shown many benefit in the scope of virtual reality. In this work, we showed that mobile interaction can benefit from similar inputs. Of course, Tangimap is more a concept than a true input device. In the next section, I present a 3 DOFs elastic controller that could be embedded in mobile devices.

1.4.2 3 DOFs elastic controller

To investigate the benefit of 3 DOFs elastic control on mobile devices, we developed a proof-of-concept prototype (see Figure 1.12). Four springs that maintain a movable grip provide the elastic feedback. The movements applied to this grip are captured by a deported target image that the device's

camera films. As part of the prototype, we also designed a tracking algorithm, which can be provided in different ways such as a driver to simulate mouse events or a dedicated library. The algorithm tracks the user's movements of the grip (in three directions) based on changes in the target image's appearance. We analyze the video frames to track the target's movements. The cardboard-made target used is a black-and-white drawing of two orthogonal axes and a central disc in its center. Additional details are described in [5].

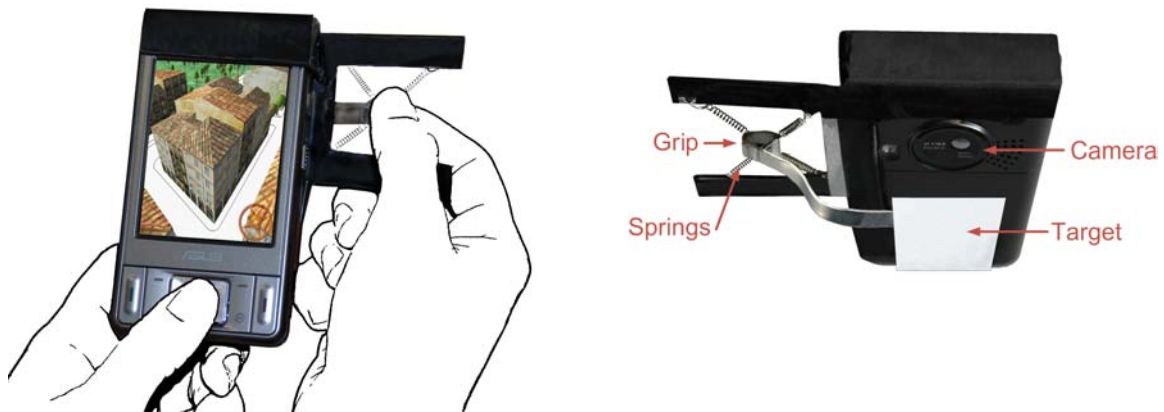


Fig. 1.12: 3 DOFs elastic controller.

Similarly to Tangimap, this elastic controller allows one to simultaneously control 3 DOFs. Consequently, it is well suited for interaction tasks that require the integral control of 3 DOFs such as interactive visualization of large 2D documents as well as manipulation and navigation in 3D environments. The main difference between Tangimap and the elastic controller is the sensing mode which the device relies on. Tangimap exploits proprioception through an isotonic sensing mode (ie. without any resistance). At the opposite, the proprioceptive feedback with the new controller is much reduced, but the inherent elastic sensing mode may be valuable for rate-controlled applications, as discussed in section 1.3.1. We did not compare the impact of these two input devices on user performances. However, we can presume that Tangimap should be better suited for position-controlled visualization of large documents while the elastic controller should better adapt to 3D rate-controlled applications.

One advantage of the 3 DOFs controller compared to Tangimap is the potential compactness of the interface for a true integration on mobile devices. We have identified two directions for this integration. The first direction consists of developing efficient physical add-ons inspired by our prototype. These add-ons will easily be adaptable to many types of mobile devices. This approach requires no electronics or mobile device modifications. Because the sensing technology is based on the embedded cameras in current mobile devices, all such devices could benefit from it. The second direction is to integrate the 3 DOFs controller directly into the mobile device. For efficient integration, the mobile device manufacturers will need to handle this work. Similar to a Swiss Army knife, the 3-DOF controller could be moved inside or outside the mobile device. This would ensure good ergonomics when the controller is used, while maintaining the handheld device's global compactness (see Figure 1.13). The elastic 3 DOFs controller we have presented lets users easily switch between the controller and another input device, for example the touch-screen for selection tasks.

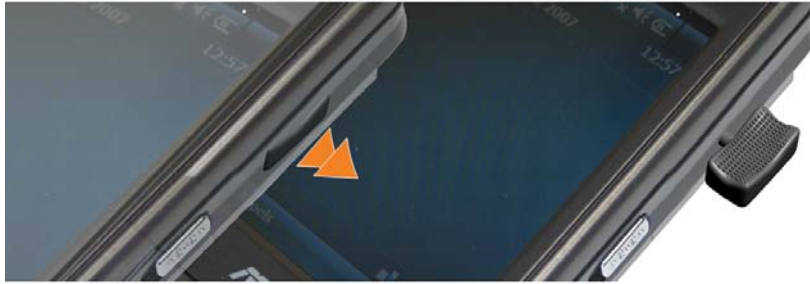


Fig. 1.13: A fully integrated solution for the 3 DOFs elastic controller (photomontage).

1.5 Conclusion

In this chapter, I presented examples of UIs designed to improve 3D interaction on mobile devices. We first focused on software interaction techniques based on the current mobile input devices, which are the physical keys and the tactile screens. Then, we explored new input modalities taking benefit from the embedded cameras of the mobile devices. I believe that many mobile 3D UIs are still to be designed to favor the development of interactive 3D graphics on mobile devices.

Beyond the input modalities we have explored, new mobile sensors could enhance mobile 3D interaction. In particular, tilt sensors, accelerometers, compasses, GPS are examples of technologies that are now embedded in the current mobile phones. Such sensors increase the mobile input space and could definitely benefit to the end-user mobile 3D applications. In addition to these devices, new technologies are emerging. This is the case of auto-stereoscopic screens that will equip the next generation of mobile devices³. Stereoscopic visualization will undoubtedly boost mobile 3D applications. It also creates new challenges for the design of efficient 3D UIs. Similarly, it can be assumed that 3D cameras such as the Z-Cam⁴ will equip mobile phones in a near future. Such a technology may motivate new approaches where users will interact by way of gestures in front or behind their mobile devices. Another direction that seems particularly interesting to me is the use of pico-projectors. Indeed, one of the main limitations of the current mobile devices is the small size of the display. Pico-projectors⁵ that now equip some first mobile devices may definitely enhance mobile interaction. Once again, UIs to be used with such distant configurations need to be developed and evaluated. Figure 1.14 illustrates some upcoming mobile technologies that may participate to the next generation of 3D mobile applications.

Tactile interaction has been the major evolution of mobile HCI during these past few years. In particular, the Iphone and its multi-touch technology has played a fundamental role in this evolution. Beyond mobile devices, touch-based interaction is changing the way we interact with digital content, from small Smartphones to large public screens. We started to focus on 3D UIs based on tactile interaction for mobile devices. Sketchball is one example. However, we believe that the benefit of touch-based UIs for interaction with 3D content goes beyond mobile configurations, and we decided to address the challenges of this new paradigm in a more general context, which is the focus of the next chapter.

³Sharp developed an autostereoscopic LCD screen that will equip the *Nintendo 3DS* and other mobile devices

⁴Z-Cam, <http://en.wikipedia.org/wiki/ZCam>

⁵eg. Samsung Pico Projector Phones I7410 and W7900



Fig. 1.14: Upcoming mobile technologies: pico-projectors, auto-stereoscopic screens, and 3D camera.

Touch-based Interaction

2.1 Introduction

General interaction with current interactive systems is evolving toward tactile interfaces. In particular, many mobile devices, tablets, interactive tables and screens tend to be operated by way of finger or stylus gestures (see for example Figure 2.1). This evolution is redefining entirely the way people interact with digital content. The past few years have seen much research activity in the scope of 2D [multi-]touch interaction (e.g. [BM08][Mos09][BWB06]). On the other hand, only very few applicative studies have leveraged the potential of touch-screens for interactive 3D applications, and research work in this area is still at an early stage. Authors have proposed sketch-based approaches for creation of 3D content from stylus input (e.g. [Iga03], [SSB08], [BBS08]). Others have explored first multi-touch 3D interaction techniques (e.g. [HCCN07][RDH09][MCG09][dIRKOD08]). I believe that a lot still need to be done in this direction.



Fig. 2.1: Interaction with a 3D object on a large touch-screen.

With mouse-based systems, users benefit from accurate pointing, distant interaction, unobstructed views of the screen, and direct access to numerous buttons and keyboard shortcuts. Consequently, the standard desktop 3D UIs have been designed from these characteristics. On the other hand, touch-screens have none of these qualities as noted by Moscovitch [Mos09]. Hence, 3D UIs for touch-screens need to be reinvented. Three years ago, we have started exploring new approaches with the goal of designing touch-based 3D UIs. We obtained first results for object inspection [14] and volume exploration [13]. We are currently exploring new 3D transformation widgets designed specifically for the purpose of touch-based interaction. In the following sections, I decided to focus Navidget, a 3D UI designed for 3D camera control. Then, I will discuss the challenges of multi-touch interaction with 3D content.

2.2 Navidget

Camera movement control in 3D applications has been studied since the early years of interactive 3D graphics. Many different user interfaces have been proposed to optimize user performance (eg. [KKS⁺05][ZF99]). However, controlling the camera in a 3D environment remains a difficult task. This is particularly true for touch-based systems where standard mouse-based approaches become inefficient [17].

2.2.1 The technique

To control camera movements in 3D touch-based applications, we have proposed a new approach called Navidget. This approach derives from the Point-of-Interest (PoI) technique introduced by Mackinlay et al. [MCR90]. Compared to a standard PoI technique where the user only selects a target, Navidget allows the control of the distance to the target as well as the viewing direction with which you want to visualize it. This is possible by way of adapted controls and visual feedback. To control the distance to the target, the user encircles the area he or she wants to focus on. The size of the focus area can be adjusted by way of some size actuators, too. A widget which is mainly composed of a half-sphere allows the user to adjust the viewing direction. By moving a 3D cursor on the surface of this half-sphere, the user controls the position of a virtual camera around the focus area. Once, the virtual camera is positioned, a smooth trajectory from the current viewpoint to the selected destination is computed and the corresponding motion starts. Navidget also integrates a border ring which allows fast access to perpendicular views. Finally, Navidget allows you to spin the half-sphere to reach the back-side of the focus area. This is done by moving the pen out of the widget and back in. Figure 2.2 illustrates the Navidget widget. Additional information including papers, demo examples, libs and videos can be found on the dedicated webpage <http://iparla.labri.fr/software/navidget/>.

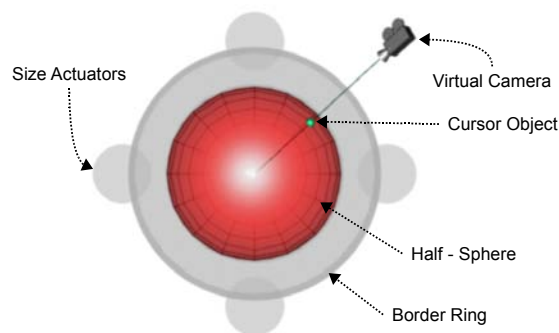


Fig. 2.2: The Navidget widget.

In addition to the described functionalities, we added some simple gestures that enrich the Navidget input space. A *back-gesture* corresponding to an upside-down "v" entails repositioning the camera in its previous state. In other words, the camera follows the last trajectory path that has been played, but in the reverse direction. From the user's perspective, this action can be seen as an "undo" action. This allows the exploration of 3D environments in a way that can be compared to website navigation, where users explore new areas of interest by way of hyperlinks, and return to previous states using

"back" actions. We think that such simple web-based navigation can help users in their navigation tasks since they benefit from a well-established mental model.

We also implemented *turn-gestures* allowing one to look around the current location. This is done by way of vertical and horizontal strokes. Following the Navidget philosophy, users do not control directly the movements of the camera. Instead, they indicate coarse directions they want to look at in simple and fast gestures. Camera movements are automatically computed once the strokes have been drawn.

2.2.2 Usages

Navidget is an example of response for our initial goal that was to enhance the mobility of users. Indeed, the same technique can be used from small mobile devices to large collaborative touch-screens. One can switch from one device to another and control complex camera movements within a unified approach, i.e. users do not need to modify the way they interact. Beyond touch-screens, we developed an immersive version of Navidget [18]. The implementation of this immersive technique operated with a virtual ray differs from its tactile counterpart. However, the approach remains the same from a user point of view. Figure 2.3 illustrates navidget operated on various systems. We also developed a remote version of Navidget, where users interact from distant mobile devices, as illustrated in Figure 2.4

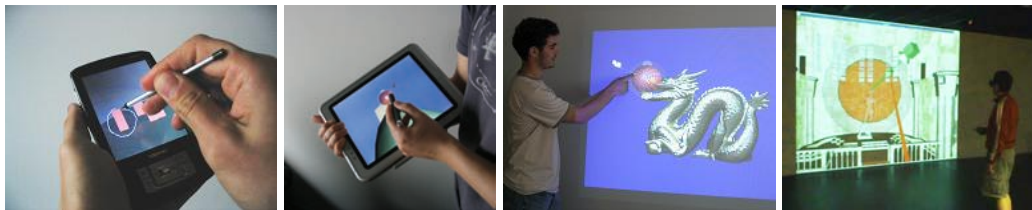


Fig. 2.3: Navidget operated on PDA, a tablet, a large screen, and in an immersive environment.

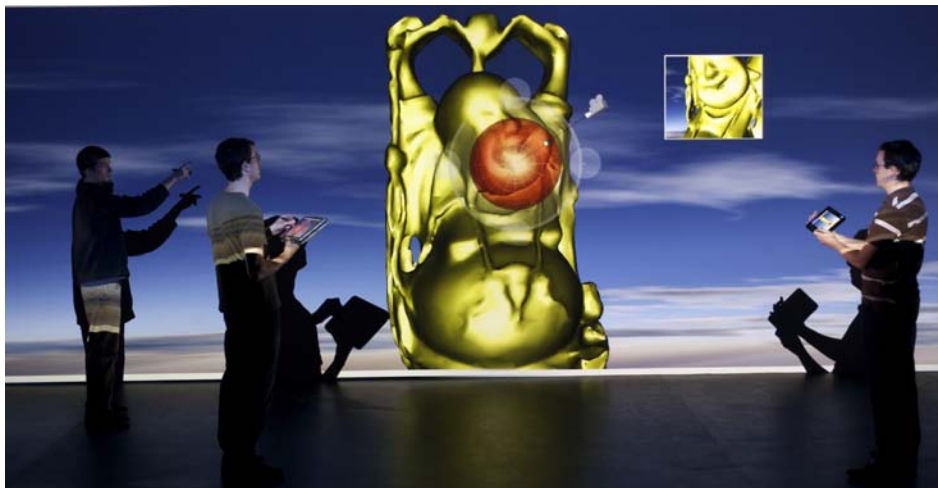


Fig. 2.4: Remote control of Navidget from mobile devices in front of a large screen.

Navidget has been demonstrated in numerous exhibitions, including Le Bourget Air Show and the European City of Science in Grand Palais, Paris. During these exhibitions, many people experimented the technique. We observed that most of the users, including kids and older people, managed to

navigate inside a 3D environment easily, and with a very reduced learning period. This success can be explained by the fact that users did not have to think about how to move the camera to reach a given goal, as done with standard approaches where several controls need to be managed (e.g. pan, orbit, and fly). Instead, they just needed to indicate what they wanted to observe in a simple and fast way, directly on the screen. The system then automatically computes the corresponding camera movements.

Navidget has also been used for collaborative applications, in the scope of Part@ge, a national ANR research project. Beyond camera control, we showed that this technique can be useful for non face-to-face 3D collaborative sessions, where users position 3D pointers from simple 2D gestures (see Figure 2.5).



Fig. 2.5: Navidget for Le Bourget Air Show (*left*), and pointers positioning for collaborative tasks (*right*).

2.3 Towards multi-touch interaction with 3D data

Generally, 3D interaction requires the control of multiple DOF. Consequently, multi-touch sensors seem very interesting compared to mono-touch technologies, for interaction with 3D content. We have explored this research direction. After a couple of years, we have observed that multi-touch technologies can benefit to 3D interaction, but the design of successful multi-touch 3D UIs is difficult and need to be addressed with a great care. In the following section, I present some of the seemingly good ideas we had. Then I will discuss the influence of directness on performance for a manipulation task from multi-touch input.

2.3.1 Seemingly good ideas

Multi-fingers interaction

Following 3D input device approaches where several DOFs can be controlled at the same time, we designed a multi-touch interface where users have access to multiple parameters from one-hand's fingers. This technique, called *Muffin* (MULTI FINGERS), is illustrated in Figure 2.6. The idea is to control 2 main DOFs with the forefinger (e.g. pan movements), while controlling additional parameters with the thumb and the middle finger (e.g. pitch and yaw camera rotations). Our experience with this technique learned us that it was very difficult for the users to assign her or his fingers to different sub-tasks. This approach may work for very simple actions, such as mouse button events as done in [LGF10], but it requires a high level of expertise for more complex tasks such as 3D viewpoint control. Consequently, we did not go forward with this approach.

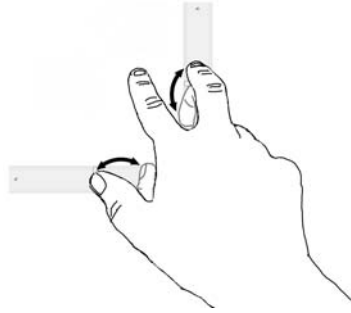


Fig. 2.6: Multi-DOF control from one hand.

Forefinger-thumb interaction

The forefinger and the thumb jointly used together enrich the input space compared to a single finger tactile input. This combination allows one to input orientations, as well as distances between both fingers in addition to 2D positions (see Figure 2.7). So, we explored this combination for the completion of 3D interaction tasks (e.g. flying vehicle metaphor). Our assumption was that the proprioceptive feedback linked to the finger relative positions may ease interaction, similarly to an elastic feedback which improves interaction on tactile screen, as I shown in section 1.3.1. Consequently, we conducted a pilot study to obtain user feedback about this approach. The main lesson of this experiment was that the subjects quickly complained because they had sore fingers due to the muscles implied during the task. Consequently, we abandoned this approach. The use of multiple fingers of one hand should be dedicated to very brief actions in 3D, as done with image resizing in 2D for example.

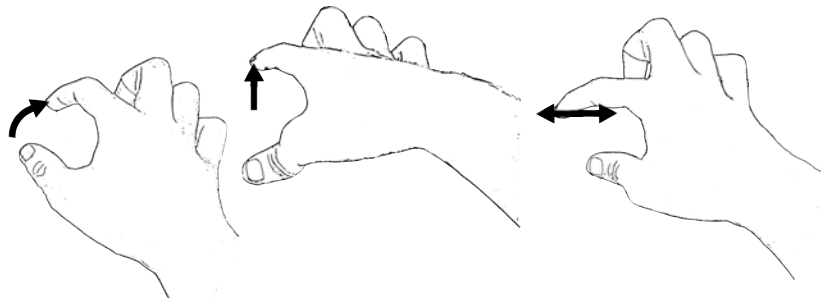


Fig. 2.7: Multi-DOF control from one hand.

Visual-Proprioceptive mapping

We have also explored proprioception when two hands are used on a distant touch surface (i.e. non co-located with the screen). Our final goal was to provide an alternative approach to the standard use of cursors for selection of visual elements on the screen (e.g. 3D widget). Our hypothesis was that the perception of the relative position of their hand may allow users to select some elements directly, without any visual feedback. For example, this could benefit to manipulation tasks where users translate objects by successively moving them along their primary axes (see Figure 2.8). To confirm this hypothesis, we first designed a low-level user study aiming at better understanding user performance in a visual-proprioceptive mapping task. Unfortunately, the results of this study were not very convincing. Hence, this has stopped our research in this direction.

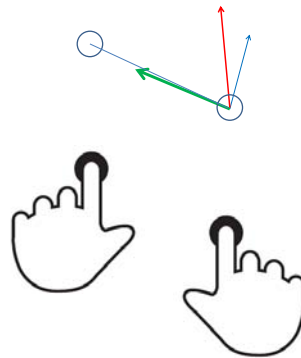


Fig. 2.8: x , y , z axes can be accessed directly from the relative position of the user's hand.

2.3.2 Extending 2D RST to 3D objects

RST is a popular multi-touch technique where *Rotations*, *Scaling*, and *Translations* are controlled from two finger inputs. With 2D content such as images, it is commonly argued that this technique is very "compelling". We have explored a similar approach for visualization of 3D objects, with the goal of understanding how well this technique extends in 3D space. With 2D visualizations, the RST technique is very straightforward because the input space and the visualization space are congruent (see Figure 2.9 left). On the other hand, a projection of an object in 3D space breaks this direct mapping (see Figure 2.9 right). Consequently, it is important to look closely at user performance for such interactive tasks.

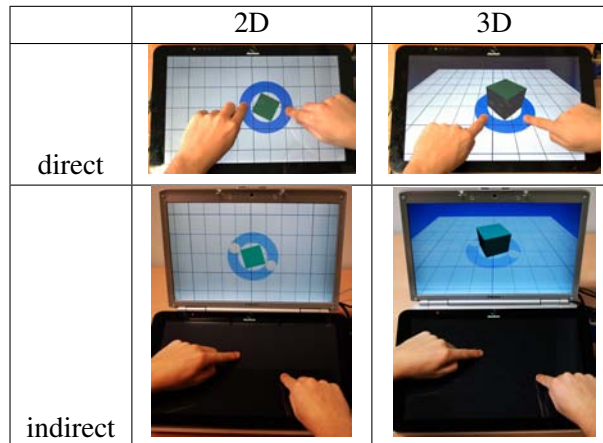


Fig. 2.9: RST technique for manipulation of photos on the Microsoft Surface (*left*). The same technique with a 3D visualization may cause occlusion problems (*right*).

In our study, we looked at the impact of directness for both 2D and 3D visualization. Indeed, indirect multi-touch interaction (e.g. users do not interact directly on the screen) has some substantial advantages that have been highlighted in the recent literature (e.g. [MH08]). In particular, indirectness avoids occlusion issues and lack of precision that are inherent to direct touch technologies. Then, contrary to its direct counterpart, indirect multi-touch design is not guided by co-location concerns. Consequently, this increases the possible variety of interactions, as user input can be mapped freely into screenspace. This seems particularly interesting for interactive 3D visualization. Moreover, indirect interaction allows one to work with large screens, or stereoscopic displays.

We designed an experiment testing the configurations that are illustrated in the following table¹:

¹In the case of 3D-indirect, the finger movements on the tactile sensors are dissociated from the screen. They are mapped to the virtual 3D plane.



24 subjects (8 women, 16 men) participated in the experiment. They were asked to complete a docking task, as fast as possible. We measured *completion time*, *target reentry* as an accuracy measure, and *inefficiency* as defined by Zhai [ZM98]. The results are illustrated in Figure 2.10².

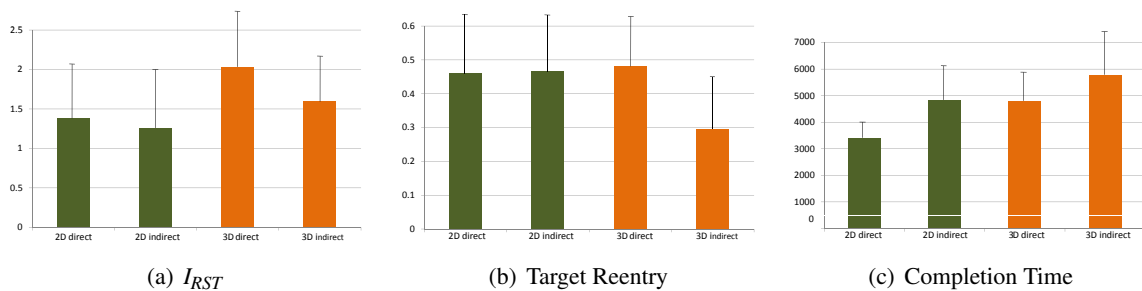


Fig. 2.10: The total mean values for inefficiency, target reentry, and completion time for all four conditions.

The important findings of this study are that although direct touch is faster, indirect interaction is more efficient and more precise for RST docking tasks. This can be explained by the fact that dissociating the visual space and the interaction space releases the co-location constraints, and consequently, let the user interact with better comfort. An analysis of the results showed that this effect was bigger in 3D than in 2D. Consequently, we believe that indirect interaction may be more suitable for manipulation of 3D data. Moreover, the results of the study showed that, with direct interaction, the speed gain is lower in 3D than in 2D. We also observed that the behaviors of the users' trajectories were very similar with 2D and 3D visualization, and with direct and indirect settings (see Figure 2.11). This tend to show that users apply similar strategies where translations and rotations are jointly managed in a coordinated way, while scaling is separated.

From these findings, we designed a demo application benefiting from RST multitouch interaction where users manipulate 3D objects on a 2D plane, from a distance (see Figure 2.12). In addition to single object manipulations, users can select and manipulate the whole scene for viewpoint adjustments. This application, dedicated to the general public, can be seen as an extension in 3D of the standard 2D multi-touch image viewer. Several users tested this demo with a furniture manipulation scenario. Interestingly, we observed that subjects with no previous experience with 3D applications and multi-touch technologies were able to arrange a virtual living room in a fast and easy way. A sim-

²We do not report all the statistical analysis in this document for clarity purpose

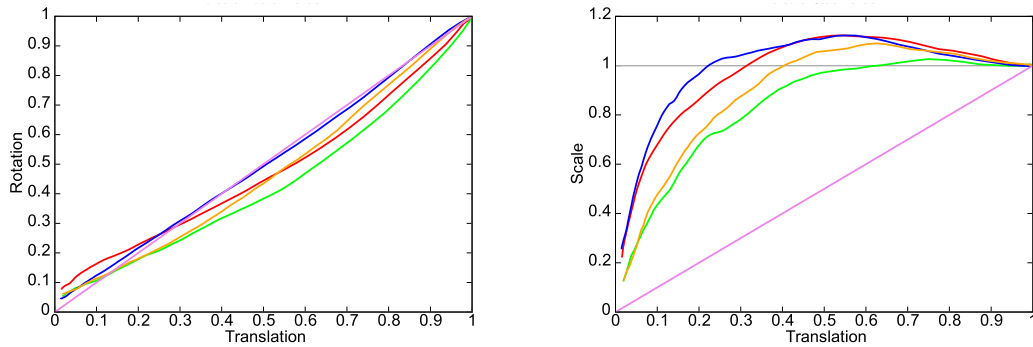


Fig. 2.11: Mismatch curves in the translation-rotation space (*left*), and the translation-scale space (*right*).

ilar task would have been difficult to complete for novice users interacting through a standard mouse approach.



Fig. 2.12: Indirect control of the RST technique in a 3D furniture manipulation scenario.

2.4 Conclusion

In this chapter, I have presented some of the work we did in the scope of touch-based 3DUI. I believe that this research direction is still promising in the future. Indeed, tactile devices become widespread. They equip an increasing number of interactive systems, and they bring users closer to digital content. This is very motivating and inspiring for the future of interactive 3D graphics, and many end-user applications can emerge from this evolution. In 2009, we have started a national ANR project, InSTInCT³, which focuses on this topic. Our goal is to explore in more depth touch-based 3D interaction, with the final goal of opening 3D graphics to non-expert users. In this project, we will continue designing 3D UIs. For example, we are currently focusing on new 3D transformation widgets. Additionally, we will explore new modalities for touch-based interaction with 3D data (e.g. stereoscopic visualization, additional sensing feedback, variety of sensor geometries and so on).

With the tactile paradigm, we have explored solutions that work from small mobile devices, to large immersive environments. Navidget is an example. This is an answer for the main objective

³InSTInCT: <http://anr-instinct.cap-sciences.net/>

of our research team Iparla, i.e. favoring the mobility of users. Beyond mobility, we also proposed solutions that bridge the gap between mobile devices and VR, where both technologies can be used together, in particular for collaborative tasks (See Sebastian Knoedel's PhD thesis [Kno09]). In the next chapter, I will discuss 3D UI for immersive virtual environments.

Immersive Interfaces

3.1 Introduction

The design of relevant 3D UIs has always been of major interest in the virtual reality (VR) community. Indeed, one of the final goal of VR is to immerse users in plausible virtual environments (See for example Figure 3.1). To achieve this goal, efficient interaction techniques need to be proposed. Since our initial investigations with the CAT in 2002, we have continued exploring immersive 3D UIs, with a special interest for artistic creation.



Fig. 3.1: A user immersed in a virtual environment.

In this chapter, I will briefly remind what the CAT is. Then, I will focus on the work we did in the scope of digital sound. This work has been mainly led by Florent Berthaut who did his PhD under the supervision of Myriam Desainte-Catherine and me. Our goal was to explore how VR could benefit to music creation. We have investigated new directions for creating new experiences.

3.2 Input devices

3.2.1 The CAT

The CAT is a 6DOF device that has been designed to favor 3D interaction in front of large projection screens. It is composed of a movable tabletop that can be manipulated around a central pillar (see Figure 3.2). The rotations (cyclic) are directly controlled by an isotonic sensing mode while the translations (infinite) are controlled by an isometric sensing mode. This allows manipulation of 3D objects in a fast and easy way, and navigation in wide virtual environments. Moreover, a tablet that is fixed on the tabletop can be used for accurate 2D interaction techniques or for the control of the system, while keeping immersed in 3D environments. Additional informations can be found in my PhD thesis [Hac03], and on the dedicated webpage: <http://www.labri.fr/perso/hachet/CAT/>. In 2003, the CAT has been used for immersive theater performances (see [32] for details).



Fig. 3.2: The CAT: Control Action Table.

3.2.2 PIIVERT

The *virtual ray* technique [BH97] is a very convincing metaphor for interacting with virtual objects. A virtual ray is generally operated by way of a 6 DOF isotonic device whose position is tracked. The user trigger events by way of low-resolution buttons. Such an interaction technique may be very efficient for many 3D interactive tasks, but it does not fit well most musical gestures, and it reduces musical possibilities. Indeed, the virtual ray metaphor suffers from accuracy issues and lack of haptic feedback. Moreover, push-buttons only output discrete events, so they cannot reproduce the dynamics of a percussion gesture. To overcome these limitations, we have designed a new input device called PIIVERT (Percussion-based Interaction for Immersive Virtual EnviRonmenTs). See Figure 3.3. Basically, with such a device, the musician will graphically select and manipulate audiovisual objects through a virtual ray metaphor (see Section 3.3.2), while using the pressure and hit gestures to interact with the selected objects within fine musical controls. This follows Cadoz’s musical gestures [Cad99], where *selection* and *modification* gestures are separated from *excitation* gestures.



Fig. 3.3: The Piivert device. Tracking is operated by infrared vision. Force Sensitive Resistors allows fine excitation from finger gestures.

Technically, we use infrared tracking for controlling the virtual ray, and Force Sensitive Resistors (FSR) for the pressure part. Figure 3.4 illustrates the setup. Note that excitation gestures do not depend from the graphical part. This ensures a high temporal precision. In addition to low-level hits that trigger sounds, we have defined high-level gestures, detected as combinations of low-level gestures. These gestures allow musicians to trigger various actions such as recording and managing musical sequences for *live-looping*¹ performances. Moreover, we extended the virtual ray metaphor

¹Live-Looping is a musical technique that consists in recording musical loops from an audio or control(e.g MIDI messages) input and stacking these loops to quickly build musical structures or sound textures.

with a *split ray* with which each finger of the Piivert device can be associated to different objects, and with a *vibrating ray*, which provides interesting feedback for percussion gestures. Additional details can be found in [10].

We conducted experiments showing that PIIVERT was more efficient than a standard virtual ray approach for musical tasks (see Florent Berthaut’s PhD thesis for details [Ber10]). Beyond musical interaction, we believe that such an approach may be valuable for other VR contexts. In particular, the high-level gestures can add semantic links between the performed gestures and the corresponding graphical actions. This work is described in more depth in [10].

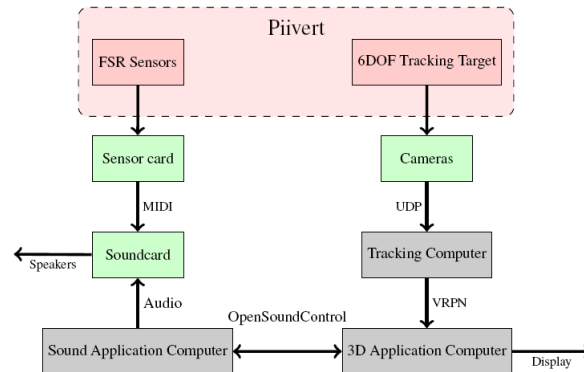


Fig. 3.4: The Piivert workflow.

3.2.3 Opportunistic music

For interaction with sounds, we have also explored an approach where musicians directly play with the physical environment surrounding them. We called this approach *opportunistic music*, following the previous work by Henderson and Feiner [HF08], where opportunistic controls were used for interaction with AR applications. The idea is to benefit from the physical attributes of the surrounding objects for controlling music. For example, a musician will slide his finger along the border of a table to control the volume of an audio track, or he will tap on the cover of a book to start a musical loop. Such an opportunistic approach opens new perspectives for musical creation. By interacting with the physical objects, the musicians benefit from inexhaustible sources of inspiration.

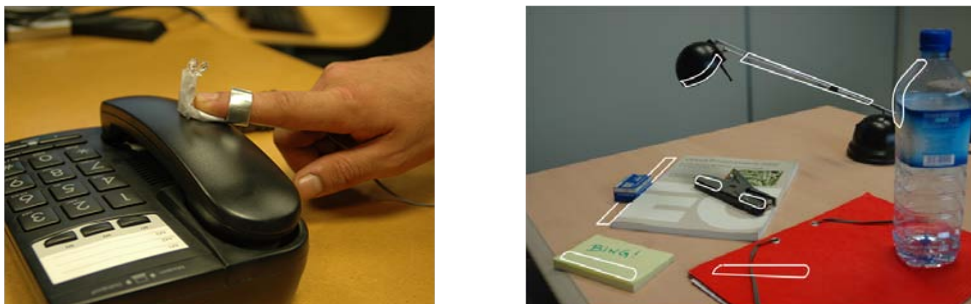


Fig. 3.5: The musician is equipped with a LED and a FSR sensor (*left*). The real environment provide many potential controllers, with interesting physical proprieties (*right*)

From a technical point of view, the musician wears a ring equipped with a FSR sensor, and a LED, which is tracked by a pair of cameras. Hence, we are able to know the 3D position of his finger, and

The second experiment focused on efficiency of single mappings in an audiovisual identification task. For each trial, an audio sequence with a variation of a sound parameter was played and four cubes were displayed, as seen in figure 3.7-a. Subjects were asked to identify as quickly as possible the cube whose graphical variations was connected to the sound parameter. The results showed almost no significant differences, but some tendencies, which may indicate that the choice of mappings scales is more important than the choice of the mappings themselves.

The goal of the last experiment was to test the influence of simultaneous audio-visual mappings. We progressively increased the number of visual feedback associated to a sound parameter to understand how user performance evolves (see Figure 3.7-b). The results indicate no significant differences, which suggest that it may be possible to combine up to four audiovisual mappings on a single graphical object without any performance loss for musicians, if they do not disrupt each other.

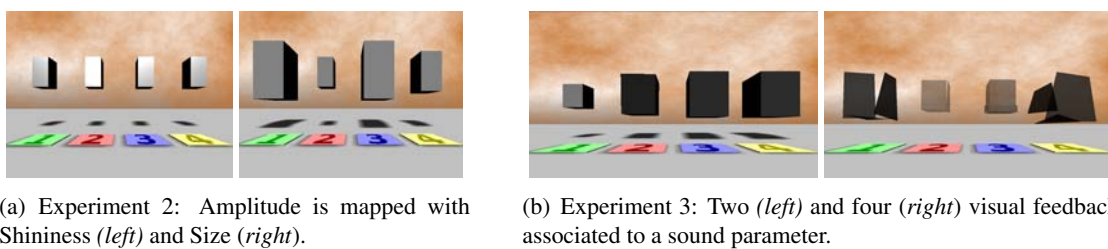


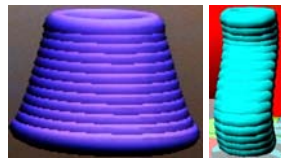
Fig. 3.7: Screenshots of the second and third experiment.

This study allowed us to better understand the interaction between interactive images and sounds. It guided us for the design of the 3D reactive widgets I will describe in the next section. The full description of the study can be found in [9].

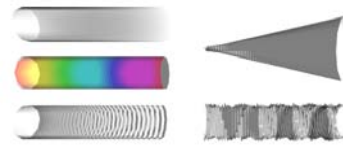
3.3.2 Interacting with the sound

We have extended the concept of *reactive widgets* to 3D. A reactive widget, described by Golan Levin [Lev00] and used for example by Sergi Jorda [Jor05], is a component offering both manipulation and visualization of musical processes. Its graphical parameters are connected to the parameters of the associated musical process. These connections are bi-directionnal, so that graphical changes are reflected in the sound process and that musical events are displayed by the widget. Figure 3.8-a illustrates examples of 3D reactive widgets. To modulate the sound, we use modification widgets as the ones illustrated in Figure 3.8-b. Hence, musicians are able to visually identify the sound they want to modulate thanks to the feedback provided by the reactive widgets. Then, they are able to modify their graphical appearance and, consequently, their audio properties by way of the modification widgets (see Figure 3.8-c). Note that both the reactive widgets and the modification widgets can be manipulated. We have also developed additional features aiming at enriching musical interaction in such 3D environments.

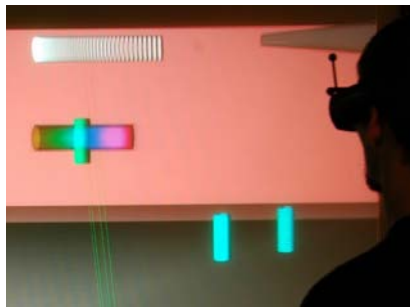
The 3D reactive widgets can be combined together, which allows the building of hierarchies. For example, bringing two reactive widgets close to each-other implies the creation of a new widget as the one illustrated in Figure 3.9. With specific Piivert high-level gestures, the reactive widgets can also be duplicated, or split. This is very interesting from a musical point of view as the musician can interact at several hierarchical levels. For example, imagine that a reactive widget has been built from initial drum sounds. Musicians may trigger only the snare or kick to produce a fill, or they can modulate the parent drum node to modify the whole rhythm. Our investigations with interactive 3D



(a) Examples of 3D reactive widgets



(b) Examples of modification widgets



(c) A 3D reactive widget being modified.

Fig. 3.8: Modifying a sound trough graphical widgets.

environments for music allowed us to develop the concept of *hierarchical live-looping*, where the interaction possibilities are greatly increased compared to the standard live-looping approaches. This new concept is described in more details in [8].

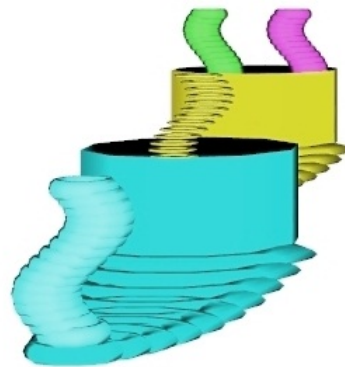


Fig. 3.9: An example of 3D reactive widget that has been hierarchically constructed from initial widgets.

3.3.3 A new virtual instrument for immersive performances

Our work in the scope musical interaction led us to the creation of a virtual instrument called Drile (see Figure 3.10). With this instrument, musicians are immersed in virtual environments displayed on large projection screens. With the Piivert device associated to head-tracking, they can interact with audiovisual objects thanks to the 3D UIs we have discussed previously, and they can build hierarchical live-looping musical structures. In addition, the musicians can move from virtual musical rooms to others by physically moving in front of the screen, and thus, they can navigate inside musical pieces. Novice

users may interact with basic components of the instrument like hit gestures and modulation gestures only. When gaining expertise, they will probably use high-level gestures to manage live-loops. After more learning, they will benefit from hierarchical structures. Videos and additional information can be found on the webpage of Florent Berthaut: <http://www.labri.fr/perso/berthaut>.

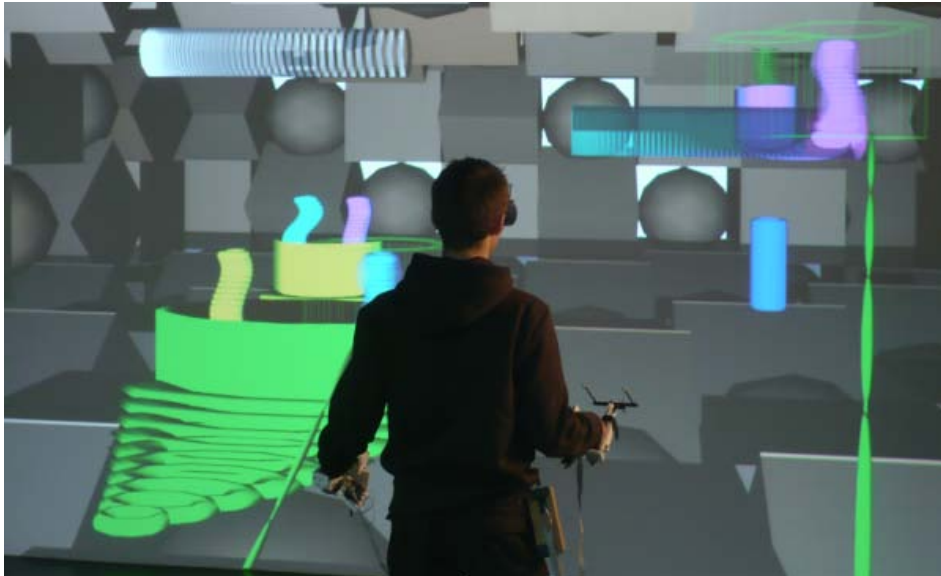


Fig. 3.10: Florent Berthaut playing music in an immersive virtual environment.

A virtual instrument like Drile allows the exploration of new musical interactions, and thus new creations, from a musician point of view. It also create new interesting experience for the audience. Because the full audience cannot benefit from head-tracking we have explored several setups to improve immersion. A configuration that has appeared as interesting is illustrated in Figure 3.11. It is composed of two screens - one for the head-tracked musician and one for the audience - corresponding to two views of the same virtual environment. With this setup, the audience benefit from a good perception of stereoscopic visualization while seeing the musicians and their gestures well. The audience see the virtual rays as if they were coming directly from the Piivert devices. They can see and understand what the musician is doing. We have performed some first performances in our lab virtual reality room with this setup. Now, we would like to exit from the lab and prepare true concerts.

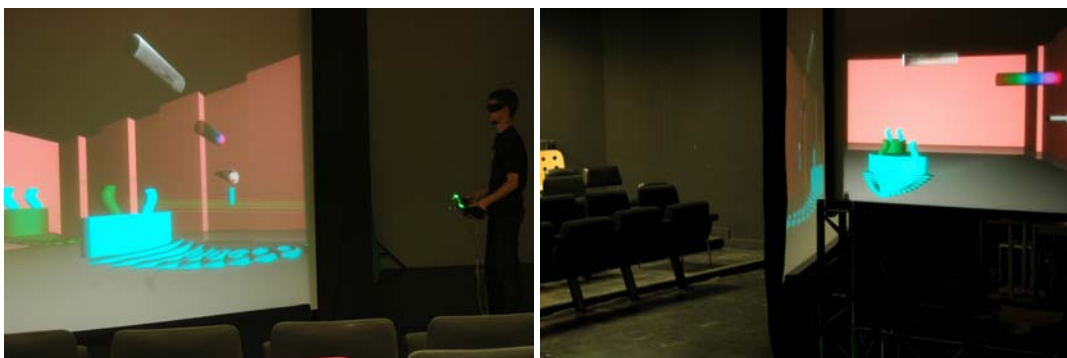


Fig. 3.11: Setup for immersive performances. The audience's view (*left*) and the musician's view.

3.4 Conclusion

The collaboration with the digital sound group has been very inspiring. When we started this work three years ago, we did not know exactly where we wanted to go. Our exploration of the possible connections between immersive 3D graphics and music creation was for me very fruitful. In one direction, the needs and constraints of music led us to the creation of new 3D UIs. Piivert is an example. We believe that this input device dedicated to music may inspire new developments for VR in general. For example, it would be interesting to explore a percussion-based approach where button inputs would be replaced by simple high-level gestures. This could add semantic to some VR applications (e.g. start an animation by successive hits of forefinger and major fingers, with the speed of the animation being proportional to the gesture velocity, and stop it with the reverse gesture). In the other direction, immersing a musician in a virtual environment led to the emergence of new musical controls that were difficult to imagine previously. In particular, hierarchical live-looping is a new musical concept we have developed from the interaction possibilities given by our interactive system. In the future, I want to continue such collaborations, and artistic creation is definitively a motivating topic.

Today, the musical interfaces we have proposed are dedicated to complex lab setups. We would like simplify them, and to make them accessible to many musicians. We hope that such simplified systems may favor new kind of artistic creations that were not possible previously. Expert musicians may benefit from such technologies. Furthermore, it would be interesting to investigate how these interfaces could motivate artistic creation for non-musician users.

Perspectives and Conclusion

4.1 Future work

In this chapter, I give some directions to the research challenges I want to tackle in the future. In the short-term, I will continue exploring 3DUI for touch-screens, as this topic is still at an early stage and a lot still need to be done. Additionally, I will start investigating another research direction that seems promising to me. This topic, whose final goal is to favor creation of 3D content for non-expert users, will be addressed in the mid-term. Finally, I will discuss additional research challenges that motivate me for the future.

4.1.1 Touch-based 3DUI, again.

In the second chapter of this document, I have summarized some first work we did in the scope of touch-based 3DUI. I think that research investigations in this direction must continue. 2D interfaces have known a mutation from mouse/keyboard systems to touch-based systems. This evolution has come with a very large emulation of research work that have addressed the challenges of such a mutation (e.g. many dedicated sessions in previous CHI and UIST conferences, and emergence of a dedicated Symposium, ITS). On the other hand, very little work have focused on touch-based 3DUI. However, all we used to do in front of our desktop computers for interaction with 3D content is not adapted to the tactile paradigm anymore. Consequently, desktop 3DUI need to be re-invented. In particular, multi-touch interaction favors the simultaneous control of several DOF, which should benefit to 3D interaction. However, our first investigations have shown that interacting well with 3D data from multi-touch input is not easy. For example, the work by Reisman et al. [RDH09] is very compelling as it allows credible manipulation of 3D objects. However, it appeared that, in practice, it is very difficult to control well the transformation being applied. Consequently, work need to be continued in this challenging research direction.

Within the InSTInCT project that I lead, the methodology we will follow relies on several steps. First, we investigate low-level characteristics for 3D interaction on touch-screens. The study of direct/indirect interaction presented in chapter 2 is an example. Currently, we also try to better understand the links between the perceived 3D data and the gestures applied to perform an action. For example, we asked subjects the gesture they would draw to make a cube spin on a touch-screen. This experiment allowed us to identify invariant behaviors of user gestures for such a task. It will be interesting to design new 3D UIs that rely on this finding. Such low-level experiments are very important because they will guide the future interaction techniques.

At the same time, we will observe how the general public tend to interact with touch-screens. One

of the partner of the InSTInCT project is Cap Sciences¹, a museum dedicated to Science. Thanks to this collaboration, we will be able to test some first techniques in true conditions. The goal is to understand some general principles that can be difficultly observed in laboratory. For example, do people use long or brief gestures? what part of the screen is touched? Do the gender and age factors impact interaction? These observations will help us for making important design choices.

Based on the low-level experiments and the observations of users interacting in true settings we will design new touch-based 3D UIs. These UI will evolve with the successive user studies we will conduct in our lab, and at Cap Sciences. In a first step, we will investigate general 3D interaction tasks (e.g. manipulation and camera viewpoint control). Then, we plan to focus on more specific tasks. For example, in collaboration with our colleagues from the SeARCH project² dedicated to cultural heritage, we will study how touch-screens may benefit to archaeologists for reassembling virtual fragments coming from scanned objects.

Beyond the design of 3D UIs dedicated to standard touch-screens, we will also try to extend the input and output spaces of touch-screens for 3D interaction. Among the directions we want to explore, we will investigate the convergence between the tactile paradigm and immersive setups. Stereoscopic visualization, additional sensing feedback (e.g. force feedback), variety of sensor geometries, are example of modalities that could benefit to 3D interaction. This will be done in particular with Immersion³, which is a company participating to the InSTInCT project.

4.1.2 Creation of 3D content for non-expert users

In addition to touch-based interaction, I plan to explore 3D UIs aiming at opening creation of 3D content to everyone. To start investigating this topic, we have just recruited Jeremy Laviolle, a new PhD student, who will be supervised by Christophe Schlick and myself.

As stated in the introduction of this document, 3D UIs for creation and modification of 3D shapes have not evolved much since the introduction of Skitters and Jacks [Bie87]. The majority of the existing modeling systems are still based on such 3D widgets, which are visual elements allowing designers to successively modify some degrees of freedom of the manipulated objects (eg. translation of an object vertex along the local-frame x axis). These modeling systems allow accurate shape creation, but they require a high degree of expertise to be used efficiently. Non-expert or untrained users cannot effectively use these interfaces; as a consequence these interfaces have not been widely adopted. More recently, authors have proposed some alternative approaches based on users' sketches to make the interactive modeling process more direct (e.g. [Iga03]). These systems exploit 2D strokes drawn by users to create 3D shapes, which can be deformed afterward [NISA07][GSMCO09]. Compared to traditional modeling interfaces, sketch-based approaches favor non-expert uses for the creation of simple shapes. On the other hand, complex shapes are hard to obtain or are even impossible to create as the range of possible modifications is limited. Finally, a common approach used for video games or for general public applications is to let users choose some variations among a predefined set of 3D objects. For example, the Creature Creator module of Spore⁴ proposes several shape variations that can be combined together in a very easy way. Some dedicated 3D widgets allow users to refine the shapes along some predefined degrees of freedom. Such an approach allows novice users to create

¹<http://www.cap-sciences.net>

²<http://anr-search.labri.fr/web>

³<http://www.immersion.fr>

⁴<http://www.spore.com/ftl>

3D objects, but the space of possible creations is limited to a few specific and simplistic interactions that are dedicated to a narrow context.

In the future, I would like to explore a novel general interaction framework where novice users would be able to define complex variations of 3D objects by way of new, simple and efficient 3D UIs. These interfaces should allow non-expert users to apply rapid coarse transformations while enabling them to precisely refine the shapes or appearances of the manipulated objects. To achieve this goal, I would like to explore an approach based on the concept of *suggestive interaction*, inspired from some computational photography tools, where the system automatically suggests in real-time various directions for the modification of objects. Such an approach allowing users to define rich variations of general 3D objects constitutes a break with the standard shape creation processes described above. It should favor successful combinations of high levels of interactivity and ease of use. In this approach, it is important to concentrate on scalable UIs, allowing taking benefit from new interaction usages, from multi-touch input to hand-free interaction.

A mixed approach based on interactive navigation in "suggestion graphs" and direct control of transformation parameters could be investigated. For example, users could be invited to "evolve" an initial object towards various transformation directions making the object thinner, smoother, more twisted or shiny. By navigating inside the suggestion graph, users should easily reach some global deformations, which may also inspire them to create shapes they would not otherwise have imagined. For each modification direction, specific widgets should be designed for precise adjustments. These context-dependent 3D widgets that can be easily understood will provide users with very specific and well-controlled operations. For example, by navigating towards a twisted variation of the initial object, the user should have direct access to widgets dedicated to the control of the twist parameters.

Many interesting questions need to be addressed to achieve efficient navigation in suggestion graphs. What are the directions that should be proposed to the users? How should these graphs be represented? Which navigation techniques must be provided? In addition to these initial questions, it will be important to understand the efficiency of such an approach from a cognitive point of view. Are users able to follow appropriate directions to reach given goals? Are they able to navigate in hierarchies while maintaining good mental models of the successive transformations? Should appearance and geometric transformations be separated or combined in the same graph? To answer these questions, we will need to design initial UIs and make them evolve according to the experiments we will conduct all along the duration of this project. Also, it would be interesting to investigate if the suggestion graphs and the dedicated widgets may adapt to the user, e.g. from predefined or learnt interaction behaviors.

Beyond suggestive interfaces, numerous 3D UIs dedicated to content creation need to be invented. To address this challenge, expertise is required in both computer graphics and 3DUI research area. In Iparla, my project-team, I have the chance to closely work with colleagues having an excellent knowledge in CG. Consequently, I think that this is very favorable for pushing such a research direction.

4.1.3 Usage-guided interaction and immersive interaction for everyone.

During the past years, my research activities have been mainly guided by technological contexts. At the beginning of the Iparla project, we have focused on 3D UIs for mobile devices. Then, we have explored 3D UIs for touch-screens. We have also explored musical interaction for immersive setups. This systematic approach has been fruitful and we have obtained interesting results. We have now gained expertise in various technological contexts. In the future, a different approach could consist

in orienting our research towards usages, independently of the technology involved. 3D applications for children or for artistic creation are target examples. Another focus could be on disability. Cultural heritage is also an example of target area where the UI will be motivated by the usage more than by the technology. For such specific contexts, we should not only focus on the design of independent 3D UIs anymore. Instead, we should address the global challenge at a higher level.

Among the emerging technologies that enhance the pool of available hardware interfaces, the immersive solutions that become available at home seems particularly interesting. In particular, 3D TV and input devices like Microsoft Kinect or Sony Move change the way people visualize and interact with 3D data. Video-games are the first targets for such new technologies. Numerous other applications could be imagined for the target area cited above. For a long time, immersive 3D UIs have been thought mainly for industrial uses. Now, the audience is changing, and new immersive 3D UIs should be invented for the general public. This new context will definitively change the way the interfaces will be developed. This is a very motivating research direction for the coming years.

4.2 Conclusion

This document synthesized the last 8 years of my research activities, where I have explored several directions for improving interaction with 3D environments. After my initial PhD work in the scope of 3D interaction in front of large projection screens, I have participated to the creation of an INRIA project-team, Iparla. An INRIA project-team focuses on one main objective during a given period (typically height to ten years). At Iparla, we chose to explore how to enhance interactive 3D graphics on mobile devices. Before starting the Iparla project-team, I did not know nothing about mobile devices. Then, by focusing our research on this topic, we have explored new approaches guided by the main objective of the team. To my point of view, this has been very fruitful, and we obtained several interesting results that are summarized in the first chapter of this document.

On the other hand, exploring a general context without having a well defined goal may lead to interesting research work, too. In particular, on the side of the Iparla's main objective, I have collaborated with the "Digital Sound" group of LaBRI. Three years ago, our initial research project was quite vague. We wanted to study the general question of how VR could enhance music creation. Then we had very interesting discussion mixing artistic and scientific considerations. These discussions led to several results that are summarized in chapter 3. Consequently, I think that having a well defined project with clear objectives as we had with the Iparla project-team is very efficient to do good research. However, at the same time, it is important to keep some time to explore new directions that are not linked to any precise objective.

3DUI in the future

The 3DUI community has been very close to the VR community for a long time. This is due to historical reasons and because the problem of how interacting well with a 3D environment is a key preoccupation in VR applications. VR is an interesting area for 3DUI. However, a confusion sometimes exists where 3DUI is seen as "VR interfaces" or 6DOF interactions. To my point of view, 3DUI is much wider than the VR specific case (and many VR topics do not concern 3DUI, e.g. virtual agents). For example, in this document, I presented examples of 3D UIs, some linked to VR, and some other not.

During the past few years, I have discussed with many people from the CG community who tend to say that many hard graphic problems were resolved, and that one of the main challenges for the coming years will be to enhance interaction between the user and 3D content. At the same time, an evolution can be noted in the HCI community where the number of publication with interfaces dedicated to 3D environments tend to increase. My feeling is that both the CG community and the HCI community that have evolved separately for a long time, are moving towards common 3DUI challenges. Consequently, I believe that 3DUI is not a narrow research area. Instead, it is a convergence topic bridging the gap between the historical communities.

Many major evolutions like printing, widespread of personal computers, or worldwide networks, have led to many advances in the scope of industry, communication, knowledge, or culture for example. I am convinced that interactive 3D graphics can play a similar role for a general evolution of our societies. Until now, this medium was mainly dedicated to expert users. I think that one of the reasons is that interaction with 3D environments remains a difficult task. In the past, many researchers have focused on the computer graphics side. I believe that, now, the main challenge is to design efficient 3D UIs that can easily be adopted by everyone.

Bibliography

- [AEY09] Roudaut Anne, Lecolinet Eric, and Guiard Yves. Microrolls: expanding touch-screen input vocabulary by distinguishing rolls vs. slides of the thumb. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 927–936, New York, NY, USA, 2009. ACM.
- [BBS08] Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. Ilovesketch: as-natural-as-possible sketching system for creating 3d curve models. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 151–160. ACM, 2008.
- [Ber10] Florent Berthaut. *Construction, manipulation et visualisation de processus sonores dans les environnements virtuels immersifs pour la performance musicale*. PhD thesis, Université Bordeaux 1, sept. 2010.
- [BH97] Doug A. Bowman and Larry F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *I3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 35–ff., New York, NY, USA, 1997. ACM.
- [Bie87] Eric Allan Bier. Skitters and jacks: interactive 3d positioning tools. In *I3D '86: Proceedings of the 1986 workshop on Interactive 3D graphics*, pages 183–196. ACM, 1987.
- [BKLP04] Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola, and Ivan Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 2004.
- [BM08] Olivier Bau and Wendy E. Mackay. Octopocus: a dynamic guide for learning gesture-based command sets. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 37–46, New York, NY, USA, 2008. ACM.
- [BWB06] Hrvoje Benko, Andrew D. Wilson, and Patrick Baudisch. Precise selection techniques for multi-touch screens. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 1263–1272, New York, NY, USA, 2006. ACM.
- [Cad99] Claude Cadoz. *Musique, geste, technologie*. Éditions Parenthèses, 1999.
- [D09] Fabrice Dècle. *Approches directes et planifiées de l'interaction 3d sur terminaux mobiles*. PhD thesis, Université de Bordeaux 1, September 2009.

- [DIRKOD08] Jean-Baptiste de la Rivière, Cédric Kervégant, Emmanuel Orvain, and Nicolas Dittlo. Cubtile: a multi-touch cubic interface. In *VRST '08: Proceedings of the 2008 ACM symposium on Virtual reality software and technology*, pages 69–72, New York, NY, USA, 2008. ACM.
- [GSMCO09] Ran Gal, Olga Sorkine, Niloy J. Mitra, and Daniel Cohen-Or. iwires: an analyze-and-edit approach to shape manipulation. *ACM Trans. Graph.*, 28(3):1–10, 2009.
- [Hac03] Martin Hachet. *Interaction avec des Environnements Virtuels affichés au moyen d'Interfaces de Visualisation Collective*. PhD thesis, Université Bordeaux 1, dec 2003.
- [HCCN07] M Hancock, S Carpendale, A Cockburn, and NZ. Shallow-depth 3d interaction: design and evaluation of one-, two-and three-touch techniques. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1147–1156. ACM, 2007.
- [Iga03] Takeo Igarashi. Freeform user interfaces for graphical computing. In *SG'03: Proceedings of the 3rd international conference on Smart graphics*, pages 39–48. Springer-Verlag, 2003.
- [Jor05] Sergi Jordà. *Crafting musical computers for new musics' performance and improvisation*. PhD thesis, Universitat Pompeu Fabra, 2005.
- [KKS⁺05] Azam Khan, Ben Komalo, Jos Stam, George Fitzmaurice, and Gordon Kurtenbach. Hovercam: interactive 3D navigation for proximal object inspection. In *proceedings of SI3D '05*, pages 73–80. ACM Press, 2005.
- [Kno09] Sebastian Knoedel. *Beyond desktop: Designing Novel User Interfaces to Enhance 3D Exploration*. PhD thesis, Université de Bordeaux 1, November 2009.
- [Lev00] Golan Levin. *Painterly Interfaces for Audiovisual Performance*. PhD thesis, 2000.
- [LGF10] G. Julian Lepinski, Tovi Grossman, and George Fitzmaurice. The design and evaluation of multitouch marking menus. In *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems*, pages 2233–2242, New York, NY, USA, 2010. ACM.
- [MCG09] Anthony Martinet, Géry Casiez, and Laurent Grisoni. 3d positioning techniques for multi-touch displays. In *VRST '09: Proceedings of the 16th ACM Symposium on Virtual Reality Software and Technology*, pages 227–228, New York, NY, USA, 2009. ACM.
- [MCR90] Jock D. Mackinlay, Stuart K. Card, and George G. Robertson. Rapid controlled movement through a virtual 3D workspace. In *proceedings of SIGGRAPH '90*, pages 171–176. ACM Press, 1990.
- [MH08] Tomer Moscovich and John F. Hughes. Indirect mappings of multi-touch input using one and two hands. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1275–1284, New York, NY, USA, 2008. ACM.

- [Mos09] Tomer Moscovich. Contact area interaction with sliding widgets. In *UIST '09: Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pages 13–22. ACM, 2009.
- [MPLKT05] Teemu Mäki-Patola, Juha Laitinen, Aki Kanerva, and Tapio Takala. Experiments with virtual reality instruments. In *Proceedings of the 2005 International Conference on New Interfaces for Musical Expression (NIME05), Vancouver, BC, Canada, 2005*.
- [NISA07] Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. Fibermesh: designing freeform surfaces with 3d curves. *ACM Trans. Graph.*, 26(3):41, 2007.
- [RDH09] Jason L. Reisman, Philip L. Davidson, and Jefferson Y. Han. A screen-space formulation for 2d and 3d direct manipulation. In *UIST '09: Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pages 69–78. ACM, 2009.
- [RGM⁺05] Xavier Rodet, Florian Gosselin, Pascal Mobuchon, Jean-Philippe Lambert, Roland Cahen, Thomas Gaudy, and Fabrice Guedy. Study of haptic and visual interaction for sound and music control in the phase project. In *Proceedings of the 2005 International Conference on New Interfaces for Musical Expression (NIME05), Vancouver, BC, Canada, 2005*.
- [SSB08] Ryan Schmidt, Karan Singh, and Ravin Balakrishnan. Sketching and composing widgets for 3d manipulation. *Computer Graphics Forum*, 27(2):301–310, 2008. Proceedings of Eurographics 2008.
- [VB07] Daniel Vogel and Patrick Baudisch. Shift: a technique for operating pen-based interfaces using touch. In *CHI, Proceedings of the 25th international conference on Human factors in computing systems*, pages 657–666. ACM, 2007.
- [WFB⁺07] Daniel Wigdor, Clifton Forlines, Patrick Baudisch, John Barnwell, and Chia Shen. Lucid touch: a see-through mobile device. In *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 269–278, New York, NY, USA, 2007. ACM.
- [ZF99] Robert C. Zeleznik and Andrew S. Forsberg. Unicam - 2d gestural camera controls for 3D environments. In *proceedings of SI3D'99*, pages 169–173, 1999.
- [Zha95] S. Zhai. *Human Performance in Six Degree of freedom Input Control*. PhD thesis, University of Toronto, 1995. http://vered.rose.utoronto.ca/people/shumin_dir/papers/PhD_Thesis/top_page.html.
- [ZM98] Shumin Zhai and Paul Milgram. Quantifying coordination in multiple dof movement and its application to evaluating 6 dof input devices. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 320–327, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co.

Personal publications

⇒ PDFs and Videos are available on my webpage: <http://www.labri.fr/~hachet>

Books Editor

- [1] Martin Hachet, Kiyoshi Kiyokawa, Joseph LaViola, Proceedings of the 5th IEEE Symposium on 3D User Interfaces (3DUI 2010) IEEE - ISBN: 978-1-4244- 6844-7, 2010
- [2] Bernd Froehlich, Ernst Kruijff, Martin Hachet, Proceedings of the 15th ACM Symposium on Virtual Reality Software and Technology (VRST 2008) -ISBN:978-1-59593-951-7- 2008

Book Chapter

- [3] Sabine Coquillart, Philippe Fuchs, Jerome Grosjean, Martin Hachet, Le traité de la réalité virtuelle, Vol. Interfaçage, immersion et interaction en environnement virtuel - Les techniques d'interaction. Presses de l'Ecole des Mines, Volume 2, page 329--380 - March 2006

Articles in journals

- [4] Martin Hachet, Fabrice Dècle, Sebastian Knödel, Pascal Guitton. Navidget for 3D Interaction: Camera Positioning and Further Uses. Int. J. Human-Computer Studies – (vol 67, issue 3), p225-236 – March 2009
- [5] Martin Hachet, Joachim Pouderoux, Pascal Guitton. 3D Elastic Control for Mobile Devices. IEEE Computer Graphics and Applications, (vol. 28, no 4), p 58--62 - July/August 2008
- [6] Martin Hachet, Pascal Guitton, Patrick Reuter, Florence Tyndiuk. The CAT for Efficient 2D and 3D Interaction as an Alternative to Mouse Adaptations. Transactions on Graphics (Proceedings of ACM SIGGRAPH), vo. 23, no 3, p 728 - August 2004 – Reprise session of VRST 2003

Additional publications in journals

- [7] Martin Hachet and Ernst Kruijff, Guest Editor's Introduction: Special Section on the ACM Symposium on Virtual Reality Software and Technology, IEEE Transactions on Visualization and Computer Graphics (vol. 16, no. 1), p 2-3 - Jan/Feb 2010

International Conferences and Workshops

- [8] Berthaut, Florent; Desainte-Catherine, Myriam ;Hachet, Martin, «DRILE: an immersive environment for hierarchical live-looping », Proceedings of New Interfaces for Musical Expression (NIME 2010) -- 2010
- [9] Berthaut, Florent; Hachet, Martin; Desainte-Catherine, Myriam, «Combining audiovisual mappings for 3D musical interaction», Proceedings of International Computer Music Conference (ICMC 2010) – 2010
- [10] Berthaut, Florent; Hachet, Martin; Desainte-Catherine, Myriam, « Piivert: Percussion Interaction for Immersive Virtual Environments », Proceedings of Symposium on 3D User Interfaces (3DUI2010)– 15–18 (4 pages) - 2010
- [11] Berthaut, Florent; Desainte-Catherine, Myriam; Hachet, Martin, «Interaction with the 3D reactive widgets for musical performance.», Proceedings of Brazilian Symposium on Computer Music (SBCM09), (2009)
- [12] Decle, Fabrice; Hachet, Martin, «A Study of Direct Versus Planned 3D Camera Manipulation on Touch-Based Mobile Phones», Proceedings of International Conference on Human-Computer Interaction with Mobile Devices and Services (Mobile HCI2009), 1--5 (2009)

- [13] Knoedel, Sebastian; Hachet, Martin; Guitton, Pascal , « Interactive Generation and Modification of Cutaway Illustrations for Polygonal Models », Proceedings of International Symposium on Smart Graphics , 140--151 (2009)
- [14] Declé, Fabrice; Hachet, Martin; Guitton, Pascal, «ScrutiCam : Camera Manipulation Technique for 3D Objects Inspection», Proceedings of the IEEE Symposium on 3D User Interfaces (3DUI) (4 pages) 19--22 (2009)
- [15] Hachet, Martin; Arash, Kian; Berthaut, Florent; Franco, Jean-Sébastien; Desainte-Catherine, Myriam, «Opportunistic Music», Proceedings of JVRC 2009 (EGVE - ICAT - EuroVR) 45—51 (2009)
- [16] Hachet, Martin; Declé, Fabrice; Knoedel, Sebastian; Guitton, Pascal, «Navidget for Easy 3D Camera Positioning from 2D Inputs», Proceedings of the IEEE Symposium on 3D User Interfaces (3DUI) 83--88 (2008) Best Paper Award
- [17] Hachet, Martin; Kulik, Alexander, «Elastic Control for Navigation Tasks on Pen-based Handheld Computers», Proceedings of the IEEE Symposium on 3D User Interfaces (3DUI2008) 91--96 (2008)
- [18] Knoedel, Sebastian; Hachet, Martin; Guitton, Pascal, «Navidget for Immersive Virtual Environments», Proceedings of 15th Symposium on Virtual Reality Software and Technology (VRST 2008), 48--50 (2008)
- [19] Zendjebil, Imane; Ababsa, Fakhr-Eddine; Didier, Jean-Yves; Vairon, Jacques; Frauciel, Luc; Hachet, Martin; Guitton, Pascal; Delmont, Romuald, «Outdoor Augmented Reality: State of the Art and Issues», Proceedings of 10th ACM/IEEE Virtual Reality International Conference (VRIC 2008) 177--187 (2008)
- [20] Hachet, Martin; Pouderoux, Joachim; Tyndiuk, F.; Guitton, Pascal, «Jump and Refine for Rapid Pointing on Mobile Phones», Proceedings of CHI 2007 (4 pages) 167--170 (2007)
- [21] Knoedel, Sebastian; Hachet, Martin; Guitton, Pascal, «Enhancing the Mapping between Real and Virtual World on Mobile Devices through efficient Rendering», Proceedings of Mixed Reality User Interfaces 2007 (IEEE VR Workshop) (2007)
- [22] Knoedel, Sebastian; Pouderoux, Joachim; Hachet, Martin; Guitton, Pascal, «3D Panorama Service on Mobile Device for Hiking», Proceedings of CHI Workshop on Mobile Spatial Interaction 106--109 (2007)
- [23] De La Rivière, J.-B.; Ditlo, N.; Hachet, Martin, «Innovative 6DOF Devices for Desktop Configurations», Proceedings of Virtual Concept électronique (2006)
- [24] Hachet, Martin; Watanabe, R.; Kitamura, Y., «A Collaborative Interface for IllusionHole using a Control-Ring and Mice», Proceedings of IEEE Symposium on 3D User Interfaces (3DUI) -Technote-, + One page (Japanese) in Proc. of the 2006 IEICE General Conference 66-68 (2006)
- [25] Hachet, Martin; Declé, Fabrice; Guitton, Pascal, «Z-Goto for Efficient Navigation in 3D Environments from Discrete Inputs», Proceedings of Virtual Reality Software and Technology (VRST) (4 pages)236--239 (2006)
- [26] Hachet, Martin; Pouderoux, Joachim; Guitton, Pascal; Gonzato, Jean-Christophe, «TangiMap - A Tangible Interface for Visualization of Large Documents on Handheld Computers», Proceedings of Graphics Interface (GI 2005) 9 - 15 (2005)
- [27] Hachet, Martin; Pouderoux, Joachim; Guitton, Pascal, «A Camera-Based Interface for Interaction with Mobile Handheld Computers», Proceedings of Symposium on Interactive 3D Graphics and Games (I3D2005), 65 - 72 (2005)
- [28] Hachet, Martin; Kitamura, Y., «3D Interaction With and From Handheld Computers», Proceedings of IEEE VR 2005 Workshop: New Directions in 3D User Interfaces électronique (2005)
- [29] Hachet, Martin; Guitton, Pascal, «The CAT - When Mice are not Enough», Proceedings of IEEE VR 2004 Workshop: Beyond Glove and Wand Based Interaction 66--69 (2004)

- [30] Hachet, Martin; Reuter, Patrick; Guitton, Pascal, «Camera Viewpoint Control with the { Interaction Table} », Proceedings of Virtual Reality International Conference (VRIC 2003) 41--47 (2003)
- [31] Hachet, Martin; Guitton, Pascal; Reuter, Patrick; Tyndiuk, F., «The CAT for efficient { 2D} and { 3D} interaction as an alternative to mouse adaptations», Proceedings of ACM Virtual Reality Software and Technology (VRST'03) 205-212 (2003). Best Paper Award
- [32] Hachet, Martin; Guitton, Pascal, «Using Virtual Reality for New Clowns», Proceedings of International Conference on Virtual Storytelling 211--219 (2003)
- [33] Hachet, Martin; De La Rivière, J.-B.; Guitton, Pascal, «Interaction in Large-Display VR Environments», Proceedings of Virtual Concept électronique (2003)
- [34] Hachet, Martin; Guitton, Pascal, «The Interaction Table: a New Input Device Designed for Interaction in Immersive Large Display Environments», Proceedings of Eurographics Workshop on Virtual Environments (EGVE 2002), Barcelone 189--196 (2002)

National Conferences (with proceedings)

- [35] Berthaut, Florent; Desainte-Catherine, Myriam; Hachet, Martin, «Widgets Réactifs 3D», Actes des Journées d'informatique musicale (2009)
- [36] Berthaut, Florent; Desainte-Catherine, M.; Hachet, Martin, «Interaction 3D pour la musique», Actes des 2emes Journées de l'AFRV 75 -- 81 (2007)
- [37] Knoedel, Sebastian; Hachet, Martin; Guitton, Pascal, «Sketch-based Route Planning with Mobile Devices in immersive Virtual Environments», Actes des 2èmes journées de l'AFRV (2007)
- [38] Declé, Fabrice; Hachet, Martin; Guitton, Pascal, «Z-Goto: pour une Navigation Efficace dans des Environnements 3D sur Terminaux Mobiles», Actes des 1eres Journées de l'AFRV 75--82 (2006)

Appendix

The appendix is composed of three publications representing each of the chapters of this document.

Mobile 3DUI:

Martin Hachet, Joaquim Pouderoux, Pascal Guitton, A Camera-based Interface for Interaction with Mobile Handheld Computers, *Symposium on Interactive 3D Graphics (I3D)*, pp. 65-71, 2005.

Touch-based Interaction:

Martin Hachet, Fabrice Dècle, Sebastian Knödel, Pascal Guitton, Navidget for 3d interaction: Camera positioning and further uses, *International Journal of Human Computer Studies*, 67(3), pp. 225-236, 2008.

Immersive Interfaces:

Florent Berthaut, Myriam Desainte-Catherine, Martin Hachet, Drile: an immersive environment for hierarchical live-looping, *Proc. of New Interfaces for Musical Expressions (NIME)*, pp. 192-197, 2010.

A Camera-Based Interface for Interaction with Mobile Handheld Computers

Martin Hachet* Joachim Pouderoux† Pascal Guitton‡
IPARLA Project, LaBRI - INRIA, University of Bordeaux



Figure 1: The camera-based interface.

Abstract

Recent advances in mobile computing allow the users to deal with 3D interactive graphics on handheld computers. Although the computing resources and screen resolutions grow steadily, user interfaces for handheld computers do not change significantly. Consequently, we designed a new 3-DOF interface adapted to the characteristics of handheld computers. This interface tracks the movement of a target that the user holds behind the screen by analyzing the video stream of the handheld computer camera. The position of the target is directly inferred from the color-codes that are printed on it using an efficient algorithm. The users can easily interact in real-time in a mobile setting. The visualization of the data is good as the target does not occlude the screen and the interaction techniques are not dependent on the orientation of the handheld computer. We used the interface in several test applications for the visualization of large images such as maps, the manipulation of 3D models, and the navigation in 3D scenes. This new interface favors the development of 2D and 3D interactive applications on handheld computers.

CR Categories: B.4.2 [Input/output and data communications]: Input/output devices—Channels and controllers; H.5.2 [Information interfaces and presentation]: User interfaces—Interaction styles

Keywords: handheld computer, user interface, interaction, visualization, video tracking

*e-mail: hachet@labri.fr

†e-mail: pouderou@labri.fr

‡e-mail: guitton@labri.fr

1 Introduction

Recent advances in mobile computing allow the users to deal with 3D interactive applications on handheld computers such as PDA and cell phones. These applications lead to new prospects for our everyday life. For example, a technician will be able to visualize and to manipulate on site the 3D model of the motor he has to repair, an archaeologist will compare the ruins of a building she saw with a computer-generated reconstruction displayed by her PDA, or a marketer will make a decision according to the strategic data he can analyze through a corresponding graph. Of course, video games have taken the benefits of mobile computing for a long time with products such as the famous GameBoy from Nintendo. They are now evolving from 2D to 3D.

Several challenges go with the development of handheld computers in terms of computing hardware, real-time visualization techniques and algorithms, and user interfaces. In this paper, we focus on this last challenge. Even if the direct pointers (stylus) seem to be widely accepted for interaction with PDA, we will see that they are not always the best solution, particularly when visualization is important. For such situations, we propose a new interface, based on the movements of a target held by the user behind the visualization interface, as illustrated in Figure 1. The target is tracked using the video stream of the camera of the handheld computer.

Compared to desktop computers, handheld computers have several

Copyright © 2005 by the Association for Computing Machinery, Inc. Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions Dept, ACM Inc., fax +1 (212) 869-0481 or e-mail permissions@acm.org.

© 2005 ACM 1-59593-013-2/05/0004 \$5.00

characteristics that have to be taken into account for the development of an adapted user interface:

- **Limited visualization.** The first characteristic is the reduced size of the screen which is one of the main limitations of the handheld computers. Consequently, the visualization area should not be occluded by the hand of the user in order to favor a good visualization. Moreover, the display of the screen can only be perceived by a limited range of viewing angles, i.e. when the user is facing the handheld computer. Therefore, interaction should operate without having to modify the orientation of the handheld computer.
- **Mobile use.** Handheld computers aim at mobile use. Consequently, interfaces for interaction have to operate in real mobile conditions. In particular, they should operate independently, without being connected to an additional computer, and without assuming the user is sitting in front of a table.
- **Limited computing resources.** The PDA and cell phones have limited computer resources. Consequently, the user interfaces have to be as less computation demanding as possible in order to insure real time interaction and to free the maximum of available resources to the applications.
- **Low extensibility.** A last characteristic that has to be taken into account is the low extensibility of the handheld computers. The limited connection possibilities and the lack of standard libraries make the integration of new I/O components difficult. Hence, the challenge for new user interfaces for handheld computers is to work on different architectures, without any custom installations.

In our approach, we attempt to keep in mind these four primordial constraints of the handheld computers. Our aim is to provide an interface that enables the manipulation of data such as 3D scenes, while assuring a good visualization. We focus on a very light user interface, in terms of bulk as well as in terms of computing resources.

This paper is organized as follows. In the next section, we present and discuss some previous work. We describe our interface in detail in Section 3, and in Section 4, we show how it can be used for 2D and 3D interaction. Finally, in Section 5, we conclude and give directions to future work.

2 Previous work

Several techniques have been proposed to interact with handheld computers. Some are based on the existing I/O possibilities, and others use new hardware interfaces for interaction. In particular, some work has been based on position and orientation sensors, and recent work uses video streams for interaction.

Buttons and direct pointers

The standard input interfaces of handheld computers are the buttons defining numerical keyboards, function keys or cursor keys for discrete input, and direct pointers such as stylus associated to sensitive screens for continuous input. They fit well with many basic interaction tasks such as numerical entries or selection of items in small menus, as experienced by Kjedskov and Skov [2002]. From these input devices, Gong and Tarasewich [2004] propose to extend the general HCI recommendations of Shneiderman [1997] to be suited to handheld computers. For higher level interaction tasks, advanced techniques have to be used. In particular, for browsing

information, the small screen size imposes the use of zooming user interfaces (ZUI) such as [Bederson and Hollan 1994] or [Igarashi and Hinckley 2000]. The main limitation of handheld computer standard devices is their limited degrees of freedom for interaction with 3D environments. Moreover, an important drawback of direct pointers is that they occlude the screens. As we recalled in the introduction, occlusion is a critical issue when dealing with very small screens.

New handheld computer interfaces

To overcome the limitation of handheld computer standard devices, several new approaches have been proposed over the last few years. Pierce and Mahaney [2003] presented an opportunistic annexing approach where the users benefit from the I/O resources available at a given moment (e.g. television and keyboard). Dusan et al. [2003] integrated speech input for multimodal interaction with handheld computers. Another example is the use of wearable chord keyboards or isometric joysticks [Krum et al. 2003].

Position and orientation sensors

Many interfaces are based on the movements of the handheld computers as input. The ScrollPad [Fallman et al. 2004] is a mouse-mounted PDA allowing the visualization of large documents while scrolling it on a table. The Peephole display [Yee 2003] is a comparable interface where the user moves the PDA in space. It is inspired from the pioneering investigations of Fitzmaurice [1993; 1993]. Recent implementations of this approach use tilt sensors [Rekimoto 1996; Hinckley et al. 2000], accelerometers and compasses [Rantakokko and Plomp 2003]. The problem of all these techniques is that the user's viewpoint to the screen changes permanently, losing the optimal visualization angle. Furthermore, the mobility is not always insured.

Cameras

The cameras of handheld computers have been used for augmented-reality applications. For example, Wagner and Schmalstieg [2003] use a PDA camera to recover the position of specific markers positioned in the real environment. Previously, Rekimoto and Nagao [1995] tracked their NaviCam by means of color-code IDs. Currently, Rohs [Rohs 2004] uses visual codes for several interaction tasks with camera-equipped cell phones. These three works are based on markers that have to be previously fixed in the real environment, limiting mobility.

Two-handed interaction

In our approach, we were inspired by previous work on two-handed interaction. Two-handed interaction demonstrated many benefits [Buxton and Myers 1986]. In particular with 3D user interfaces, two-handed interaction has been used to give the users a kinaesthetic reference frame [Balakrishnan and Hinckley 1999]. The principle is to perform actions with the dominant hand with respect to the non-dominant hand corresponding to the reference frame. For example, Mine [1998] proposed several interaction techniques for immersive virtual environments using two hands in order to benefit from the kinaesthetic sense (proprioception). Our interface exploits kinaesthesia as well.

3 A new camera-based interface

3.1 General description

Inspired by the forces and the weaknesses of the described previous work, and according to the recommendations we presented in the introduction, we developed a new interface for interaction with handheld computers.

To favor a **good visualization** we opted for a system where the screen is never occluded by the users' hand. Consequently, the users can concentrate on their data without being perturbed by any physical object. Moreover, we particularly wanted the users not to be constrained to move the handheld computer for interaction, letting them choose and keep the optimal viewing angle to the screen. In our system, the user interacts by doing movements behind the handheld computer.

Mobile use is the aim of handheld computers. Therefore, we developed a system that can be used everywhere at any time, without being connected to any desktop computer, even through a wireless network. To overcome the problem of **low extensibility**, we use the camera that is more and more often integrated in the handheld computer, or that can easily be plugged in the CF or SD ports. The inputs for interaction infer from the analysis of the images coming from the camera.

To deal with the **limited computing resources** of the handheld computers, we designed a target that will be described in the next section. By simply analyzing a few pixels of the video stream, we are able to quickly determine the 3D position of the target with respect to the handheld computer. Our algorithm requires very little CPU ticks, hence freeing the computing resources for the applications.

Summing up, the users interact with the data by moving a target behind the screen as illustrated in Figure 2. The target has approximately the size of a CD jacket. We detect the position of the target in 3D space, which makes our interface a 3-DOF interface. This approach takes benefit from the two-handed interaction and the kinesthetic sense described above.



Figure 2: The camera of the handheld computer films the movements of a target behind the screen.

3.2 Implementation

Computer vision toolkits such as OpenCV¹ or ARtoolkit² track a target in real time on standard PC. For example, Woods et al. [2003] use the ARToolKit to emulate a 6-DOF flying mouse. Similarly, Hinckley [1999] developed a camera-based multi-DOF mouse for pose estimation from a 2D grid pattern. However, the use of such techniques on handheld computers is not reasonable as they are very CPU demanding. Furthermore, using a wireless connection to process the vision algorithms on a distant server as done by Wagner and Schmalstieg [2003] is critical since the response of the user's action depends on the QoS of the network. Moreover, such a technique is only possible when a wireless network is available. Therefore, we designed an efficient and fast algorithm that can be directly computed on the handheld computers.

The 3-DOF tracking of a single black pattern on a white target is simple to implement. However, the small field of view of the handheld computer cameras makes this technique inefficient since the target has to be far from the camera to be visible while moving. Moreover, the input images have to be entirely analyzed, which significantly increases computation time.

In order to favor wide movements and to reduce the number of operations for tracking, a second approach could be the use of color scales. Hence, a x - y location of the target could be directly inferred from the color of the pointed pixel. Several color spaces could be used, in particular the uniform CIE Luv color space. However, the low quality of the handheld computer cameras and the variation of the light conditions in mobile settings make this technique inappropriate.

RGB color-codes

In order to allow wide movements close or far to the camera, and to limit the issues due to the variable light conditions, our technique is based on pure RGB color-codes. We use the target shown in Figure 3 composed of several cells separated by red borders. The target is a 12×12 cm wide square divided into $8 \times 8 = 64$ cells. Each cell is composed of two horizontal lines. The upper line relates to the x coordinate while the lower one relates to the y coordinate of the target. Each of these two lines is composed of a triplet of blue and green colors. By assigning 0 to blue and 1 to green, each triplet corresponds to a binary code. Consequently, in our target, the first line of any cell codes the column number in the target while the second line codes the line number.

A pixel from the video input is defined by its three RGB components. Each of the pixels analyzed by our algorithm is assigned to one of the three clusters RED, GREEN or BLUE according to the maximum of its components.

In the general case, it is very simple (and fast) to determine which location on the target the camera is pointing to. We describe the technique in the following. The *seed* is a reference point that is initially set to the center of the input image. The snapshots on the right are parts of the input images.

¹OpenCV: <http://www.intel.com/research/mrl/research/opencv>

²ARToolKit: <http://www.hitl.washington.edu/artoolkit>

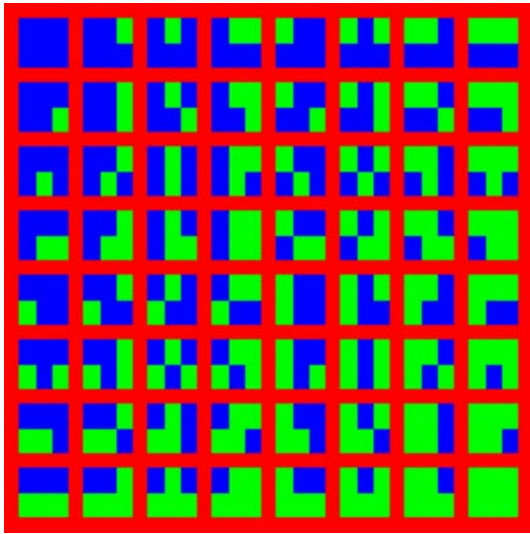
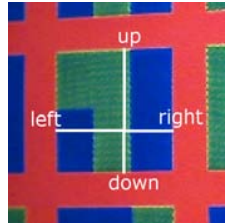
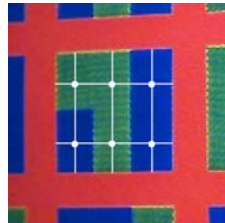


Figure 3: The RGB target composed of binary code cells.

From the seed, we search for the borders of the cell to the right, to the left, to the top and to the bottom by following the simple rule: *while the pixel is not red, consider the next pixel.*



Once the borders of the cell have been identified, it is very easy to recover the color-codes by looking at the six pixels highlighted on the right.



The color-codes determine which cell is pointed to. In our example, the first line of the cell has as the binary code 110, which corresponds to the sixth column. The second line is 010, corresponding to the third line.

The location pointed by the center of the camera has the following x coordinate on the target:

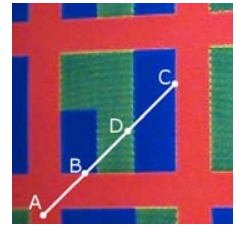
$$x_{target} = N * W + l_s * \frac{W}{l_r} \quad (1)$$

where N is the column number, W is the width of the column on the target, l_s corresponds to the number of pixels between *left* and *seed*, and l_r is the number of pixels between *left* and *right*. The y_{target} coordinate is computed identically.

The x - y position of the target with respect to the camera is directly given by x_{target} and y_{target} . The z coordinate is inferred from l_r , the number of pixels between *left* and *right*. Indeed, the farther the target is from the camera, the smaller is the distance *left to right*.

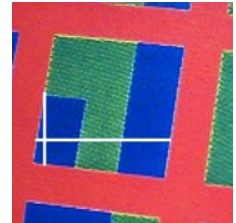
We have described a technique to determine the position of the target when the seed belongs to a cell. When the seed does not, it has to be shifted to the next cell as follows.

A is the initial seed. *While the current pixel is red, go to the next up-right pixel.* Set B . *While the current pixel is not red, go to the next up-right pixel.* Set C . The new seed D is defined by the middle of BC .

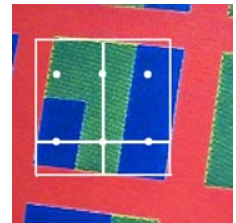


To make our technique more robust, a last test is performed before looking for the borders of the cell. This last test detects the error that could come from a tilt of the target.

Since the cells are squares, the lengths *right-left* and *up-down* are supposed to be equal. If not, we are in the situation illustrated on the right.



In this particular case, we shift the seed to the middle of the longest segment between *right-left* and *up-down*. Applying this heuristic allows to deal with small tilts, which is enough from our experience.



The technique described here operates well. The users are able to perform wide and fine displacements of the target, far or close to the camera. These displacements are processed by our algorithm and provide x , y , and z coordinates for each frame. These coordinates are absolute since they do not depend on the previous records. When a record is too far from the average of the 3 previous records, it is considered as an error and not taken into account.

3.3 Technical considerations

For the first tests, we use a PocketPC Toshiba e800 PDA with a 400MHz XScale ARM CPU. With such a PDA the x , y , and z coordinates can be estimated in less than one quarter of millisecond. Consequently, our approach allows a real-time interaction and can be used without penalizing the application. Thanks to its efficiency, our interface could be used with smaller CPU such as cell phones.

We use the Compact Flash FlyCAM camera with a resolution of 160×120 pixels. As noticed in [Wagner and Schmalstieg 2003], this camera is not able to deliver more than 7-8 images per seconds, which constitutes the main bottleneck today. However, we assume that more and more efficient integrated cameras will equip the PDA and the cell phones, and that the video streams will be easily exploitable.

Once folded, the target can be put in the cover of the PDA.

For the 3D applications we are going to describe in next section, we use *klimt*³, an OpenGL—ES implementation for PocketPC.

³klimt: <http://studierstube.org/klimt>

4 Applications

Many applications can benefit from our interface, for the visualization of large documents (e.g. maps, pictures and web pages) or for interaction with 3D environments. In the following, we present some test applications we have developed.

4.1 Pan and zoom

The visualization of large documents such as maps is a difficult task when dealing handheld computers. Our 3-DOF interface is particularly well adapted to such a task since the users are able to directly control pan and zoom, without occluding the screen. They just have to bring the target closer in order to focus on a specific location. They bring it farther for general views. The two-handed interaction and the kinaesthetic reference frame allow an efficient interaction as the users benefit from a spatial feedback in addition to the visual feedback. Figure 4 illustrates the use of our interface for the visualization of a large map.



Figure 4: Large map visualization.

4.2 3D interaction

The following interaction techniques show how our interface can contribute to the development of 3D applications on handheld computers. These interaction techniques relate to manipulation, navigation, selection, and system control.

Manipulation

Our interface provides 3-DOF that can easily be attached either to the translations or to the rotations of a 3D object according to the pressed button. The mapping between the translations of both the held target and the manipulated object is direct. It gives the users the feeling to directly hold the object. Consequently, users are able to translate the objects in 3D space without any difficulties. Rotating an object is less direct as the user's translations are mapped to the rotations of the object. However, all the users who tested the system immediately understood the technique without any explanations, because the semantic link between the users' actions and the resulting actions on the screen operates well. The mapping between the movement of the target and the rotation of the object is illustrated in Figure 5.

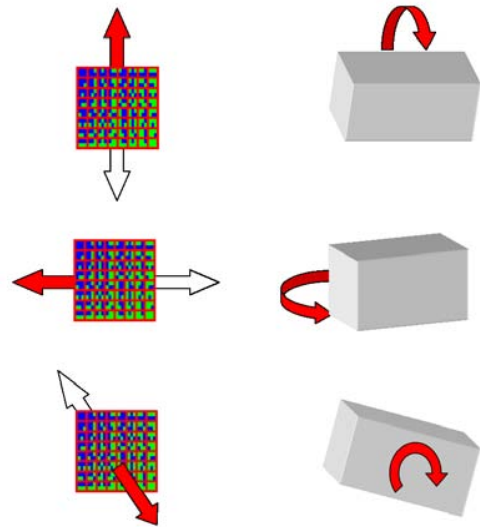


Figure 5: Mapping between the target movements and the object rotations.

Navigation

The 3-DOF provided by the interface could be used to first control the translations of the camera, and then its rotations. However, we preferred to implement a technique to directly fly into the 3D scene by using the *forward-backward* DOF to control the *forward-backward* velocity in the scene and the 2 other DOF to modify the *yaw* and *pitch* angles. Hence, by controlling 3 DOF at the same time, users are able to easily navigate in large 3D scenes. To go to the left, they move the target to the left. To go up, they move it up. Finally, to go slower, they just have to bring the target closer. Once again, the visualization is not altered as the user permanently sees the whole screen over the best orientation. Figure 6 shows an example of such a 3D navigation in a height field created using AutoMNT [Pouderoux et al. 2004].

Selection and system control

Several interaction techniques can be used to select 3D objects. For example, a technique could be to use the target as a butterfly net, or to extend the classical 2D box selection to 3D space. Therefore, users would be able to select groups of objects by locking them up into 3D selection areas. This approach is used by Encarnação et al. [1999] with their translucent sketchpad.

The control of the application is a real problem when dealing with handheld computers. Effectively, the small size of the screen forbids the use of large popup menus, as classically done on standard screens. Therefore, new user interfaces have to be found. We propose to use treemaps that hierarchically refine to select the menu items. The level of hierarchy is controlled by the *z* DOF while the *x* and *y* DOF enable to navigate over the items for a given level. For example, we can imagine that at the first level (when the target is far from the camera) five items divide the treemap: "file", "edit"... By bringing closer the target in the "file" section, eight new items appear, and so on. After a small moment, users know where the



Figure 6: Height field navigation.

items are on the target allowing them to quickly control the system. For example, the item "save as" will be on the top left of the target.

5 Conclusions and future work

Numerical data have been visualized on computer monitors for a long time. Today, handheld computers also allow mobile visualization which leads to many prospects in our everyday life. However, the intrinsic characteristics of the handheld computers make us think about new user interfaces.

The classical input devices of the handheld computer (buttons and direct pointers) can be very efficient for some basic interaction tasks. However, they become obsolete for the visualization and/or 3D applications. That's why we developed a new 3-DOF interface for interaction with handheld computers. This interface aims at favoring a good visualization at any time, in real mobile settings, with real low computing resources. Our interface based on a simple video stream analysis works well and the users who tested it were really enthusiastic. They particularly like to permanently see the whole screen when navigating in 2D maps or 3D environments. Moreover, they really enjoyed the smoothness of the 3D trajectories they were able to control.

The next step of our research project will consist in evaluating the interface. In particular, we will ask subjects to perform a given task with our interface and with a 2D direct pointer. For example, the task could be a 3D docking task or 3D labyrinth navigation task. Our interface has also been implemented on a standard PC using a web-cam. It can be a low-cost alternative to classical 3D interfaces. We have to evaluate the benefit of this approach for such environments.

In this paper, we presented a new interface for 2D and 3D mobile

applications. Farther than these applications, we think that this interface has a great potential for information visualization (InfoVis) applications such as the visualization of large graphs. Similar to the application for the visualization of pictures that we have presented, InfoVis applications would benefit from two-handed interaction and kinaesthetic reference frame for an efficient interaction. Many InfoVis techniques, such as Fisheye, could be used. We are currently focusing on these techniques.

Acknowledgements

The work described in this paper was supported by a grant from the Conseil Régional d'Aquitaine, France.

References

- BALAKRISHNAN, R., AND HINCKLEY, K. 1999. The role of kinesthetic reference frames in two-handed input performance. In *Proceedings of the 12th annual ACM symposium on User interface software and technology*, ACM Press, 171–178.
- BEDERSON, B. B., AND HOLLAN, J. D. 1994. Pad++: A zooming graphical interface for exploring alternate interface physics. In *Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology (UIST'94)*, 17–26.
- BUXTON, W., AND MYERS, B. 1986. A study in two-handed input. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, 321–326.
- DUSAN, S., GADBOIS, G., AND FLANAGAN, J. 2003. Multi-modal interaction on pda's integrating speech and pen inputs. In *Proceedings of EUROSPEECH*, 2225–2228.
- ENCARNAO, L., BIMBER, O., SCHMALSTIEG, D., AND CHANDLER, S. 1999. A translucent sketchpad for the virtual table exploring motion-based gesture recognition. In *Proceedings of Eurographics '99*, 179–190.
- FALLMAN, D., LUND, A., AND WIBERG, M. 2004. Scrollpad: Tangible scrolling with mobile devices. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 9 (To appear)*.
- FITZMAURICE, G. W., ZHAI, S., AND CHIGNELL, M. H. 1993. Virtual reality for palmtop computers. *ACM Trans. Inf. Syst.* 11, 3, 197–218.
- FITZMAURICE, G. W. 1993. Situated information spaces and spatially aware palmtop computers. *Communications of the ACM*.
- GONG, J., AND TARASEWICH, P. 2004. Guidelines for handheld device interface design. In *Proceedings of DSI 2004 (To appear)*.
- HINCKLEY, K., SINCLAIR, M., HANSON, E., SZELISKI, R., AND CONWAY, M. 1999. The VideoMouse: a camera-based multi-degree-of-freedom input device. In *UIST '99: Proceedings of the 12th annual ACM symposium on User interface software and technology*, ACM Press, 103–112.
- HINCKLEY, K., PIERCE, J. S., SINCLAIR, M., AND HORVITZ, E. 2000. Sensing techniques for mobile interaction. In *Proceedings of the 13th annual ACM symposium on User Interface Software and Technology (UIST 2000)*, 91–100.

- IGARASHI, T., AND HINCKLEY, K. 2000. Speed-dependent automatic zooming for browsing large documents. In *Proceedings of the 13th annual ACM symposium on User Interface Software and Technology (UIST 2000)*, ACM Press, 139–148.
- KJEDSKOV, J., AND SKOV, M. B. 2002. Interaction design for handheld computers. In *Proceedings of the 5th Asian Pacific Conference on Human-Computer Interaction, APCHI 2002*, Science Press.
- KRUM, D. M., MELBY, R., RIBARSKY, W., AND HODGES, L. 2003. Isometric pointer interfaces for wearable 3D visualization. In *CHI '03 extended abstracts on Human factors in computing systems*, ACM Press, 774–775.
- MINE, M. 1998. *Exploiting proprioception in virtual-environment interaction*. PhD thesis, University of North Carolina at Chapel Hill.
- PIERCE, J., MAHANEY, H., AND ABOWD, G. 2003. Opportunistic annexing for handheld devices: Opportunities and challenges. Tech. rep., Georgia Institute of Technology GIT-GVU-03-31.
- POUDEROUX, J., GONZATO, J.-C., GUITTON, P., AND GRANIER, X., 2004. A software for reconstructing 3d-terrains from scanned maps. ACM SIGGRAPH 2004 Sketches and Applications, <http://iparla.labri.fr/software/automnt/>.
- RANTAKOKKO, T., AND PLOMP, J. 2003. An adaptive map-based interface for situated services. In *Proceedings of Smart Object Conference*.
- REKIMOTO, J., AND NAGAO, K. 1995. The world through the computer: Computer augmented interaction with real world environments. In *ACM Symposium on User Interface Software and Technology (UIST 1995)*, 29–36.
- REKIMOTO, J. 1996. Tilting operations for small screen interfaces. In *ACM Symposium on User Interface Software and Technology (UIST 96)*, 167–168.
- ROHS, M. 2004. Real-world interaction with camera-phones. In *2nd International Symposium on Ubiquitous Computing Systems (UCS 2004)*.
- SHNEIDERMAN, B. 1997. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley Longman Publishing Co., Inc.
- WAGNER, D., AND SCHMALSTIEG, D. 2003. First steps towards handheld augmented reality. In *Proceedings of Seventh IEEE International Symposium on Wearable Computers*.
- WOODS, E., MASON, P., AND BILLINGHURST, M. 2003. Magic-mouse: an inexpensive 6-degree-of-freedom mouse. In *Proceedings of the 1st international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, ACM Press, 285–286.
- YEE, K.-P. 2003. Peephole displays: pen interaction on spatially aware handheld computers. In *Proceedings of the conference on Human factors in computing systems*, ACM Press, 1–8.

Navidget for 3D interaction: Camera positioning and further uses

Martin Hachet*, Fabrice Declé, Sebastian Knödel, Pascal Guitton

INRIA - Université de Bordeaux, France

Received 21 April 2008; received in revised form 22 September 2008; accepted 23 September 2008

Available online 11 October 2008

Abstract

This paper presents an extended version of Navidget. Navidget is a new interaction technique for camera positioning in 3D environments. This technique derives from the point-of-interest (POI) approaches where the endpoint of a trajectory is selected for smooth camera motions. Unlike the existing POI techniques, Navidget does not attempt to automatically estimate where and how the user wants to move. Instead, it provides good feedback and control for fast and easy interactive camera positioning. Navidget can also be useful for distant inspection when used with a preview window. This new 3D user interface is totally based on 2D inputs. As a result, it is appropriate for a wide variety of visualization systems, from small handheld devices to large interactive displays. A user study on TabletPC shows that the usability of Navidget is very good for both expert and novice users. This new technique is more appropriate than the conventional 3D viewer interfaces in numerous 3D camera positioning tasks. Apart from these tasks, the Navidget approach can be useful for further purposes such as collaborative work and animation.

© 2008 Elsevier Ltd. All rights reserved.

Keywords: 3D camera control; Pen-input; 3D widget; Collaboration; Animation; 3D pointer

1. Introduction

Navidget is a 3D user interface (3DUI) designed for easy camera positioning in 3D environments using 2D inputs. This technique was initially published in 3DUI 2008 (Hachet et al., 2008). This paper is an extension of the previous publication. In particular, we propose new functionality to better control the camera. We also show that the Navidget approach can be useful for other interaction tasks, such as collaborative work and animation.

Camera movement control in 3D applications has been widely studied since the early years of interactive 3D graphics. Many different user interfaces have been proposed to optimize the user performance. However, controlling the camera in a 3D environment remains a difficult task, and innovative interaction techniques still need to be designed to make user interaction easier.

In this paper, we propose a new UI attempting to reach this goal. This interface, called Navidget, allows easy and fast camera positioning in 3D environments from 2D inputs, by way of an adapted widget. Navidget derives from the *point-of-interest* (POI) technique introduced by Mackinlay et al. (1990), where the user selects the endpoint of a trajectory in order to automatically fly to a corresponding location. This technique is also known as the “go to” function for a wide range of 3D viewers. Selecting the endpoint of a trajectory as a navigation metaphor has several advantages:

Generic inputs. Only a pointing device and a start signal are requested by POI techniques. No additional standard workstation devices, such as keyboards or joysticks, are needed. Consequently, a POI approach can be used with a wide variety of alternative equipments, including mobile devices (e.g. smartphones and PDAs) and large touch screens for collective work. Indeed, the possible inputs with such equipments are often limited to pointing operations (as illustrated in Fig. 1). Consequently, POI techniques are particularly well suited for optimizing 3D navigation in alternative visualization situations.

*Corresponding author. Fax: +33 5 40 00 66 69.

E-mail addresses: hachet@labri.fr (M. Hachet), declé@labri.fr (F. Declé), knoedel@labri.fr (S. Knödel), guitton@labri.fr (P. Guitton).

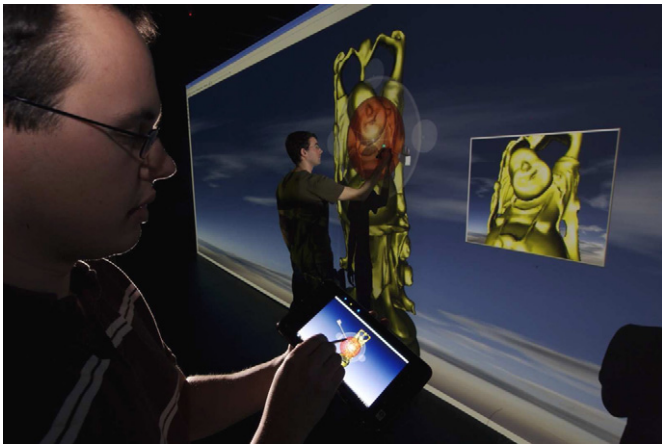


Fig. 1. Navidget on a UMPC and a large screen. ©CNRS Phototheque / C. Lebedinsky.

Ease-of-use. POI approaches are very simple and direct. Users simply have to select the point to which they want to fly. No learning time is required. All visible parts of the scene can be reached directly.

Fast. Camera movement in the 3D environment operates within controlled completion times. Hence, users can quickly travel long distance.

Cognitive-friendly. Cognitive maps, which play an important role in wayfinding tasks, are sometimes difficult for the user to construct when navigating in large 3D environments. POI users benefit from continuous movements (i.e. the whole environment is traveled over from the current position to the desired location). Moreover, the speed of camera movement gives an indication of the distance traveled.

Real-time independent. True real-time cannot always be achieved in 3D applications, particularly when very large scenes have to be displayed or when low computation devices such as handhelds are used. In this case, direct camera controls must be avoided, since the time-lag between the user's actions and the related camera movement in the 3D scene can be very disturbing. With POI approaches, the user simply selects a target and waits for the camera movement, so the time-lag is much less objectionable as an immediate feedback is not required.

On the other hand, POI techniques have two significant drawbacks:

Surface-dependent. The target is always a point on a surface. Consequently, the only possible camera movements are translations towards a surface of the 3D scene. This is restrictive as users may want to move to other locations. For example if two objects are displayed, users have to choose the one they want to fly to. It is not possible to move to a location where both objects can be seen within a closer view.

Limited control. Users can only give the point to which they want to fly. They do not control how they want to go there. In particular, users do not control parameters such as the distance between the endpoint of the camera

trajectory and the target, and the angle from which they want to view the target. These parameters are generally fixed or automatically computed. This results in some awkward situations. For example, facing a large surface that occludes the whole scene is very common when using classical POI techniques.

In spite of these two drawbacks, POI techniques are still widely used as they are often the easiest techniques. “Go to” is one of the standard functions of Web3D viewers. In many cases, users navigate in 3D environments by simply pointing to some part of the scene. POI techniques are useful in many visualization contexts for novices as well as for expert users. In this paper, we propose to extend the POI approach by providing additional controls to the user.

In the next section, we present some previous work that are related to this approach. We then describe the Navidget technique in detail in Section 3. In Section 4, we present the user study that has been carried out. We present how Navidget has been extended in Section 5. In addition to camera positioning tasks, we show in Section 6 that the Navidget approach can be beneficial to the user for many other tasks. Finally, in Section 7, we give our conclusion and suggest some directions for future work.

2. Previous work

Control of the virtual camera in 3D environments requires at least 6 degrees of freedom (DOF). These DOF can be directly controlled by means of 6-DOF input devices using adapted metaphors. For example, Ware and Osborne (1990) proposed the *scene-in-hand*, *eyeball-in-hand*, and *flying vehicle* metaphors. However, such 6-DOF devices are not that common. Consequently, the challenge that researchers have tried to resolve for many years is how to control the camera using only the 2 DOF available on conventional input devices, such as mice.

With 2-DOF devices, the standard techniques are *orbiting*, *flying* and POI. Orbiting is used for observation of an object from an exocentric point of view. The principle is to map the motions of the controller to motions over the surface of a sphere (Chen et al., 1988), or other 3D primitives. 3D motions can also be achieved by means of external widgets, as is the case with Open Inventor-like viewers (Strauss and Carey, 1992). Other advanced techniques such as the HoverCam (Khan et al., 2005) and the StyleCam (Burtnyk et al., 2002), are based on camera movements along predetermined surfaces or trajectories. Orbiting techniques are mainly used for observing objects.

The flying approach, which is widely used in video games, is related to the *flying vehicle* egocentric metaphor, but with only 2 DOF. These 2 DOF are generally assigned to forward/backward movements and left/right camera rotations. If a keyboard is available, the forward/backward movements can be controlled using certain keys, leaving up/down 2D controller movements free to control top/bottom camera rotations. Some advanced techniques allow

automatically constraining camera trajectories according to the topology of the scene (e.g. Hanson and Wernert, 1997), or according to the speed of the movements (Tan et al., 2001). Flying techniques are mainly used for exploring 3D scenes. Contrary to the above techniques, Navidget does not directly map the input DOF to the camera DOF. It rather allows the user to define target destinations, without having in mind the camera DOF that are implied.

The POI approach has been shown to have many benefits, as discussed in the introduction. This technique has primarily been described by Mackinlay et al. (1990), where logarithmic control of speed is proposed. Some extensions have since been explored. For example, in Mackinlay et al. (1994), radial and/or lateral viewpoint motion have been discussed. Zeleznik and Forsberg (1999) describe the UniCam where one of their techniques, called *click-to-focus on silhouette edges*, is aimed at automatically choosing the endpoint of the camera trajectory according to the proximity of the edges of some object. Hachet et al. (2006) proposed the *z-goto* for mobile devices, an extension of the “go to” approach, where the endpoint is directly selected in depth by means of simple keystrokes. For all these techniques, the virtual camera is automatically positioned in the 3D space. Users do not have any direct control once the target point is selected. In our approach, we want the user to be able to control the camera destinations. Zeleznik et al. (2002) have investigated the use of controlled POI for VR setups. They proposed a set of techniques where users can adjust the viewing direction and the distance to the target by way of tracked VR devices. Navidget shares the same goal, but for standard 2D screen-based interfaces.

In the scope of computer graphics, 3D widgets such as Bier’s (1987) Skitters and Jacks are commonly used. Today, the wide majority of the modeling applications are based on these visual 3D elements for the manipulation of 3D objects (e.g. Blender). These widgets have shown many benefits for manipulation tasks (translate/rotate/scale). On the other hand, the use of 3D widgets for navigation and inspection tasks has been little explored. The IBar (Singh et al., 2004) has been proposed to control some parameters of the camera in an interactive fashion (e.g. perspective distortion). In our approach, we donot want to control the camera parameters independently. We rather provide an interface that allows the user to complete a 3D camera positioning task in a unique operation.

An alternative approach for 3D interaction tasks is the use of sketch-based techniques. The idea is to let the user control the system from simple 2D strokes. For example, SKETCH (Zeleznik et al., 1996) and Teddy (Igarashi et al., 1999) interpret the input gesture for modeling and editing tasks. A sketching philosophy has also been proposed for animation tasks in *Motion Doodles* (Thorne, 2004). Concerning camera manipulation, Unicam (Zeleznik and Forsberg, 1999) is based on a gesture vocabulary for the control of the camera parameters. Finally, the system

proposed by Igarashi et al. (1998) allows 3D walkthrough from free-form path drawing on the 3D scene ground. These gesture-based techniques, where the user can control the system by way of simple 2D gestures inspired our work.

3. Navidget

3.1. General approach

The main idea of Navidget is to let the users control where they want to look at, contrary to the conventional POI techniques that try to guess where the user wants to focus. With a 2D sequence only (i.e. press, move and release), the Navidget users define their target destinations. This is done using adapted controls and a dedicated widget.

3.2. Navidget controls

3.2.1. Distance

The simplest gesture sequence that is related to Navidget consists in pointing a target destination by picking a 3D point in the scene. This is similar to the conventional POI approach. The main problem with this basic approach is that the camera can only be moved towards a surface. Moreover, the resulting view may be little convincing as the user does not control the depth where the camera has to stop (see Fig. 2). To solve these problems, we propose an initial extension consisting in circling the target area.

Circling is highly intuitive as the user directly draws what he or she would like to see, as illustrated in Fig. 2. Circling is particularly well suited to stylus-based and finger-based systems. Indeed, this gesture is usually done with a pen on a paper when something has to be highlighted. The circling metaphor has been used by Schmalstieg et al. (1999) with transparent props for selecting virtual objects. However, their work did not involve navigation tasks. Circling can be achieved using any pointing device, depending on the hardware settings. With mouse-based environments, the circling metaphor can be replaced by the conventional rectangle-based selection technique, which is commonly used in many 2D software programs or with some 3D orthographic CAD applications.

After a release event, the camera moves towards the center of the circle. There are several possibilities to compute the final camera position, as presented in Fig. 3. The first and simplest one (CENTER) makes use of the 3D point given by the projection of the circle center into the

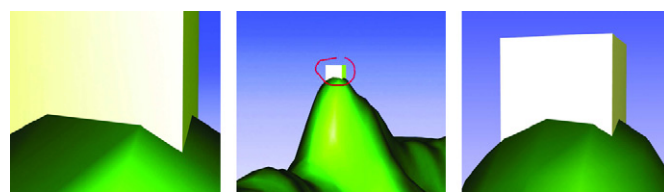


Fig. 2. Pointing a target may induce senseless views (left). Circling it moves the camera to an appropriate view (right).

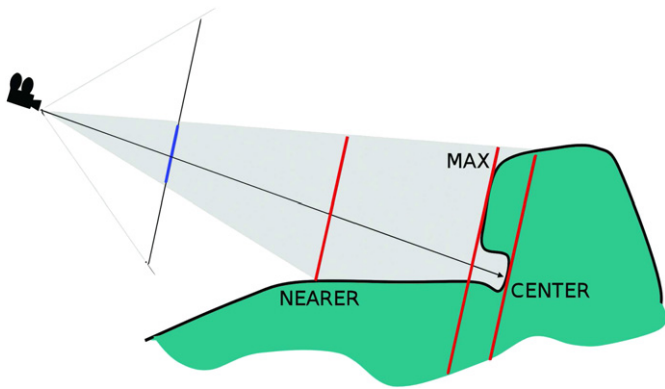


Fig. 3. The three different possibilities (NEAR, MAX, CENTER) to compute the depth value.

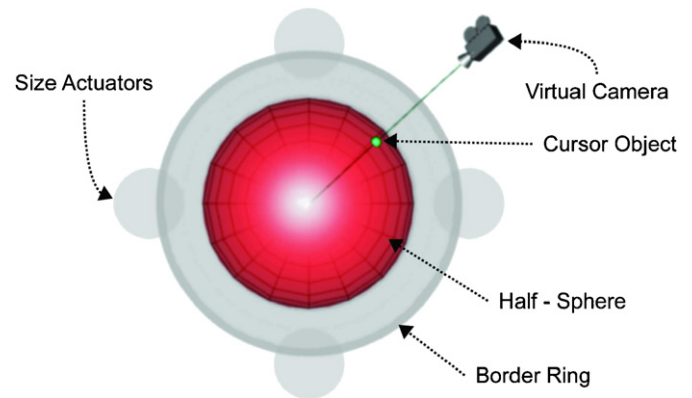


Fig. 4. The Navidget.

scene. The drawback of that naive approach is that this particular point does not necessarily exist. To overcome that issue, we developed an alternative approach (MAX) based on the most frequent depth value of the selected area. The main drawback of this method is that objects that are located between the current viewpoint and the target area might not be visible afterwards. Finally, the most conservative technique (NEARER) considers the nearest depth value of the selected area in order to insure that the whole selection is visible.

Because of perspective, the resulting view is not exactly what has been circled. It should be noticed that Zeleznik and Forsberg (1999) have proposed a concept of *region zooming* that may be related to our technique. However, with their technique, users must first select a point before modifying the size of the zooming area. Consequently, the target area is always centered on a 3D point in the scene. We think that circling is a very intuitive metaphor, as users directly show what they want to focus on using a simple and familiar gesture.

3.2.2. Viewing direction

The targeted viewing direction is automatically computed in existing POI techniques. Generally, the camera looks at the target point with a viewing direction that is aligned with the normal of the related face, or with a pre-defined oblique view (Forsberg et al., 1998). Whatever the choice made by the designers of POI techniques, the final camera angle is fixed. Consequently, users cannot control how they want to visualize the target. Worse, automatic computation of the destination views can be disturbing. For example, the viewpoint can be moved to a position which is too close to the target.

We use a 3D widget in order to let the user control the viewing direction at destination. This widget, illustrated in Fig. 4 is composed of a half-sphere facing the current viewpoint, a border ring, a set of size actuators, and a virtual camera. This widget is centered on the target. It appears if no release events occur just after a pointing operation. In other words, if the user clicks and waits for a short time, the widget appears. At this level, a default

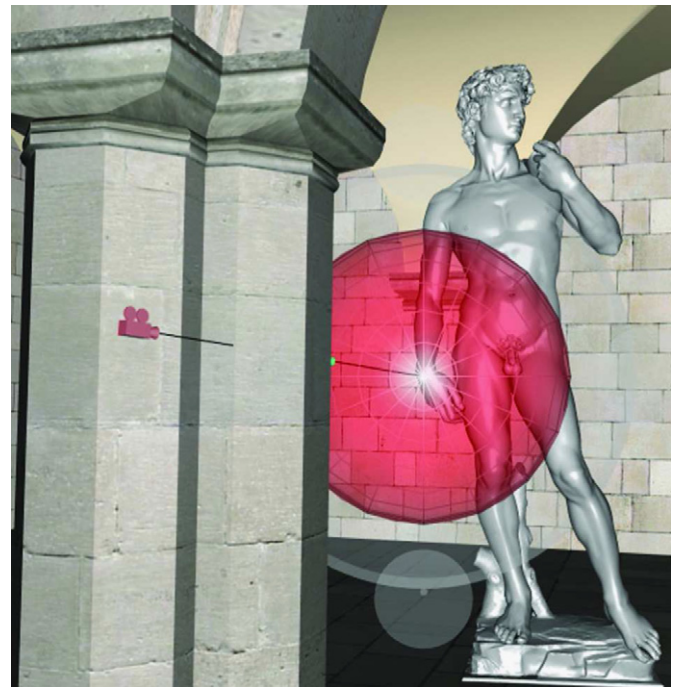


Fig. 5. The visual feedback provided by Navidget allows the user to avoid occluded views.

radius is computed in order for the widget to appear as a constant proportion of the application window. We did not base our approach on object bounding-boxes in order not to constrain the user interaction to object hierarchies. Indeed, Navidget allows the users to focus on specific parts of large objects such as terrains, or to travel towards areas where several objects are implied.

The half-sphere highlights the part of the scene that will be seen at the end of the camera trajectory. By moving a 3D cursor on this surface, users control how they want to visualize the target. A representation of a camera shows the position and angle to which the viewpoint will be moved. This intrinsically provides good feedback to the user. Indeed, the possible occlusions can be seen clearly, so they are naturally avoided (see Fig. 5). In Section 3.2.5, we present an extension, called *smart camera*, which allows

emphasizing the visual feedback that is provided by Navidget.

A pilot study that will be described in Section 4 revealed a need for efficient selection of viewpoints that are orthogonal to the current viewing direction. For this purpose, we added a border ring. Indeed, reaching the border ring is more convenient than accurately positioning the cursor on the border of the half-sphere. Moving the cursor around the border ring has similar effect than moving it along the visible limits of the half-sphere.

The pilot study also revealed that the users wanted to move the virtual camera behind the half-sphere. Consequently, we introduced a mechanism allowing switching the direction of the half-sphere, in front or behind the target depth plane. When the cursor exits and re-enters the half-sphere, the latter is 180° rotated through a quick animation (see Fig. 6). This allows moving the virtual camera all around the target area. A fast outside–inside movement allows going directly behind the target area. A similar gesture within a slower movement insures continuity in the virtual camera movements, the border ring being similar for both front and back half-spheres.

3.2.3. Distance and viewing direction

Similarly to the pointing technique, the widget appears if no release event occurs once a circling gesture has been performed. After a hold phase, the widget is displayed and the radius is directly given by the size of the drawn circle. In this way, the user can manage both the distance to the target and the final viewing direction. This sketch-based approach enhances intuitive interaction. Of course, the widget can be called after rectangle-based area selection, too.

If available, the barrel button, the non-preferred hand, pop up menus or pressure-based input could also be used to make the widget appear. However, since our technique is based on a simple and unique gesture sequence by a one handed interaction, we want to avoid extra pop up menus as well as additional interaction using the non-preferred hand. Furthermore, using the barrel button or a pressure based input is not suitable for stylus or finger-based interaction system.

We added some size actuators that allow the widget to be resized at any time, without releasing the stylus or the finger. These actuators are located on the cardinal direction

of the half-sphere. Once these actuators are captured, the system is set to a RESIZE state. The user can then modify the size of the widget—and consequently the distance of the virtual camera to the target—by doing vertical (resp. horizontal) movements on the screen. The RESIZE state is left when horizontal (resp. vertical) movements are detected (see Fig. 7). This means the user wants to come back to the half-sphere or the border ring to specify the viewing direction. Hence, by controlling a direction and a distance, users are able to position the camera in the 3D space.

When a release event occurs and the cursor is located on the half-sphere or the border ring, the viewpoint is smoothly moved to the target virtual camera. If the cursor is outside, the action is canceled and the widget is hidden.

To obtain a smooth camera movement we create a key-frame animation by interpolating the position and orientation of the initial viewpoint and the specified target camera. After a circling gesture we use a simple linear interpolation, which results in a straight-lined movement towards the selected target. If the half-sphere widget was used to specify the target, we perform a Bézier interpolation that results in a smooth curved trajectory. Every animation lasts 100 key-frames (ca. 3 s), which guarantees that the target is reached in the same time, independently of the distance to travel.

3.2.4. Preview window

Navidget has initially been designed to favor fast camera positioning in 3D environments. We noticed that it could also be useful for distant inspection tasks. A preview window corresponding to the virtual camera's view can be displayed in addition to the main visualization window (see Fig. 8). Thereby, users can inspect distant areas and decide whether or not to move to the corresponding targets, according to what they have seen. Concretely, if the users just want to inspect objects without moving in the scene, they will release the pen outside of the widget. This refers to a cancel action. If they want to move to the target, they will release inside the widget, once the preview visualizations correspond to their targets.

It can be noticed that this technique recall previous work such as the one by Grosjean and Coquillart (1999), where distant inspection was possible from manipulation of a 6-DOF magic mirror in immersive virtual environments.

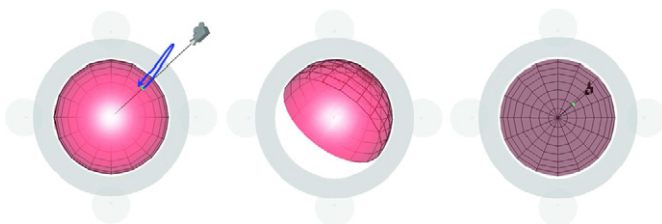


Fig. 6. An outside–inside movement (*blue stroke*) allows switching the orientation of the half-sphere (front or back), so the virtual camera can be moved behind the target area.

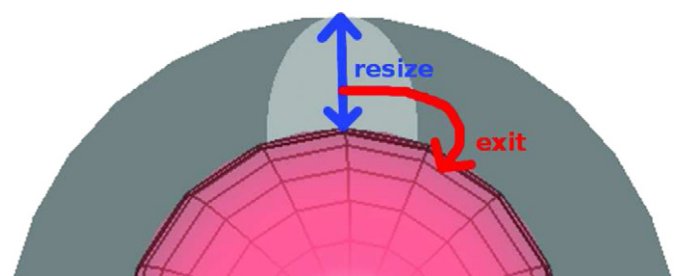


Fig. 7. The size actuators.

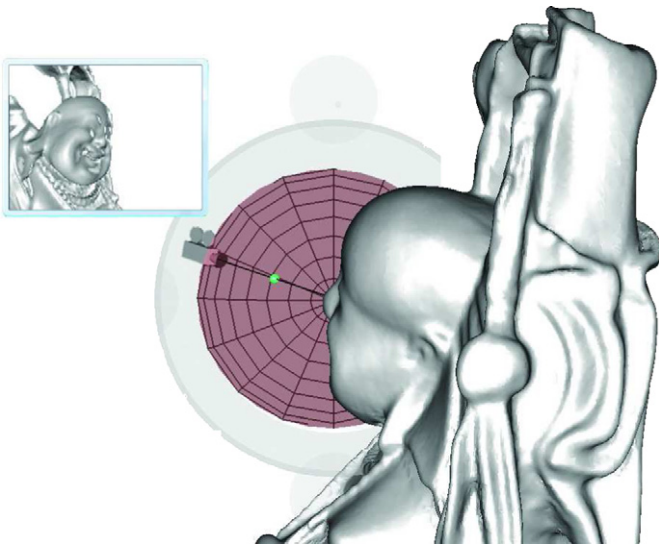


Fig. 8. The preview window. In this example, the half-sphere has been moved behind the target area.

With Navidget associated with a preview window, the user can look all around the target area with a 2-DOF input control.

A preview window is not adapted when small displays such as those of the mobile devices are used. Additional windows may also be disturbing with immersive stereoscopic visualization. However, when the display of an additional window is possible, the use of a preview window with Navidget can be very useful. This allows exploring a 3D scene without moving one's position. This is particularly interesting as the user can focus on specific areas while keeping the global context. This *focus + context* approach is useful for the cognitive processes that are implied in 3D navigation tasks. It has to be noticed that this technique requires additional computation. Consequently, real-time rendering can be a problem with big 3D scenes.

3.2.5. Smart camera

As we have already described in Section 3.2.2, Navidget provides users with good visual feedback. In addition to this intrinsic feedback, we propose an extension that supports the user by providing additional information as the camera is being positioned. This extension is useful as soon as an object is located between the POI and the position of the virtual camera, or when the camera is covered by an occluder. Both cases are presented in Fig. 9(a) and (c).

Since we know the position of the POI and the current position of the camera, we are able to detect any geometry between these two positions that could disturb the user. Therefore, we can visualize the occluding part in high contrast (see Fig. 9(b)) so it is clearly visible to the user. Furthermore, we can prevent the camera from vanishing behind an object by making any such objects transparent, as illustrated in Fig. 9(d).

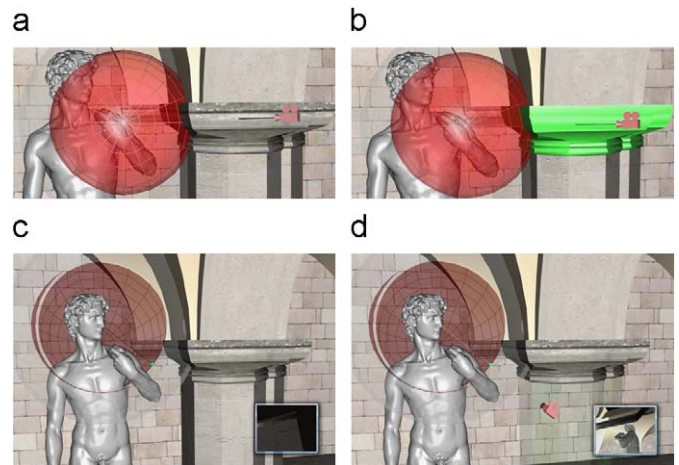


Fig. 9. Smart camera: (a, c) without smart camera and (b, d) with smart camera.

It should be noted that the computational cost to detect occlusions increases with the complexity of the scene. Indeed, precise occlusion computation implies numerous intersection tests. In order to preserve real-time, we currently use a simple method based on two intersection tests. The first one tests the visibility from the virtual camera to the center of the half-sphere (Fig. 9(b)). The second one tests the visibility between the current view-point and the virtual camera (Fig. 9(d)).

Beyond these tests, we could also move the camera to a more convenient position where no occlusions occur. However, this would imply taking away control from the user, which may cause disturbances.

4. User study

4.1. General comments

In this section, we describe a pilot study we carried out to assess Navidget. We have designed a set of experiments in order to evaluate the general usability of the technique. The feedback we obtained from the subjects who participated in this study helped us for the final design of Navidget. For all the experiments, a TabletPC was used. Our study focused on tasks where Navidget can be useful, i.e. when the user wants to visualize 3D areas from specific viewpoints.

Our motivation for this study was to evaluate the usability of each Navidget control. Moreover, we wanted to have an idea about the user preference between Navidget and the standard 3D viewer controls.¹ We set up experiments where the subjects had to complete camera positioning tasks. After the experiments, we asked the subjects to answer a questionnaire.

In our preliminary experiment, we did not use time as a metric to evaluate Navidget in comparison with the standard controls. Indeed, completion times largely depend on the speed that is chosen for Navidget animations. It also

¹Standard controls were *fly*, *pan*, *look around*, *go to* and *orbiting*.

depends on the sensibility levels that are set for the direct controls. Consequently, the completion times obtained would have been strongly linked to the initial settings.

4.2. Procedure

Thirty subjects divided into two groups took part in this first study (21 males, nine females, average age = 26). For the first group, 15 3D expert users were recruited. The second group included 15 novice users (i.e. having less than one experience using videogames or 3D applications a month). For each group, half of the subjects completed all the experiments with Navidget, then with the standard controls. The other half began with the standard controls.

The experimental scenes consisted in cubes for which one face was highlighted. On this face, some letters (red “A” and black “B”) were written. The tasks entails counting the number of red “A”. To complete the task, the subjects had to move to appropriate viewpoints (see Fig. 10). A trial was completed when the subject selects a number as an answer of the task. This was done using an answer panel.

Before the experiments, we presented both interfaces to the subjects. They tried each control at least once inside a test environment to get familiar with them.

During the experiment with Navidget, the subjects were explicitly asked to use one of the Navidget controls (i.e. *point only*, *circle only*, *point + widget*, *circle + widget*, or *widget + size actuators*). Each control has been used three times with different targets’ size and location.

4.3. Results

The analysis of the questionnaires have shown that the general usability of Navidget is good. Both expert and novice subjects did not have difficulties when using the Navidget controls (see Table 1). The circling metaphor has been appreciated. Indeed, the link between circling and focusing is very strong. Consequently, the circling metaphor was used extensively by the subjects, even if a simple pointing technique would have been necessary. Similarly, sketching the widget by way of a circle gesture works well. The link between the drawn circle and the corresponding 3D widget was easily understood by the subjects. The subjects managed to define the targeted viewing direction easily. They have understood the role of the half-sphere

directly. Similarly, the subjects have controlled the size actuators efficiently, from the first use.

Among the comments we had, some subjects complained not to be able to move the virtual camera behind the half-sphere. For this reason, we have developed the mechanism that enables to switch the half-sphere front and back. Some additional subjects have tested this mechanism in an informal way. They appreciated the technique and they have used it without difficulties.

Similarly, the border ring has been developed from subjects’ comments. The addition of this functionality has improved the interface. Indeed, side views can be reached very fast. We also have good feedbacks from the first users concerning the Navidget preview window.

The pilot study has shown that the wide majority of the subjects preferred using Navidget rather than the conventional controls for the proposed tasks, as shown in Fig. 11.

Wilcoxon signed rank tests have shown that this preference was significant for each question we asked² to both experts and novices, except for the ease of understanding where both interfaces were easily understood. In particular, both expert and novice subjects found that Navidget was easier to use than the standard controls (Expert *Wilcoxon Z test*: $-3.52, p < 0.05$; Novice *Wilcoxon Z test*: $-2.99, p < 0.05$). They also found Navidget faster (Expert *Wilcoxon Z test*: $-3.45, p < 0.05$; Novice *Wilcoxon Z test*: $-3.34, p < 0.05$).

With Navidget, a simple action is used to complete the task. The same task requires switching between different techniques when using the standard controls, which is time consuming. Following Buxton (1995), each control switch divides the action in separate chunks, because the applied motor tension changes and the motion continuity is interrupted. This leads to higher separation into subtasks and therefore to higher cognitive load.

4.4. Discussion

Novice users had some difficulties using conventional controls they were not familiar with, particularly since they had to remember the specific nature of each of them (translation in the xy plane means nothing for a novice user). Generally, they were trying a control without knowing what was going to happen in the 3D scene. They often got lost. Some of them also reported some motion sickness.

On the other hand, with Navidget, the subjects enjoyed dealing with a technique where several parameters can be controlled by a simple gesture. Unlike the standard controls where the 2 input DOF are directly mapped to 2 of the camera viewpoint DOF (e.g. z -translation + y -rotation for *fly*), the 2D gestures of Navidget make it possible to perform basic camera positioning tasks, where multi-DOF are jointly implied. The rich visual feedback of Navidget favors efficient interaction. During



Fig. 10. Example of an experimental scene for the pilot study. From the initial view (left), the subjects had to move closer to the target face (right) in order to count the number red “A”.

²See Fig. 11 for the questions.

Table 1
Satisfaction for each of the controls of Navidget [Novices, Experts (mean)]

	Circle	Half-sphere	Resize actuators	Circle + Half – sphere
Was the interaction technique easy to understand?	3.0, 2.9 (3.0)	2.9, 2.8 (2.8)	2.6, 2.6 (2.6)	3.0, 2.9 (3.0)
Was the interaction technique easy to use?	2.9, 2.9 (2.9)	2.7, 2.7 (2.7)	2.2, 2.4 (2.3)	2.7, 2.8 (2.8)
Were you able to interact freely?	2.9, 2.9 (2.9)	2.5, 2.9 (2.7)	2.3, 2.8 (2.5)	2.9, 2.9 (2.9)
Did you feel like interacting precisely?	2.9, 2.6 (2.7)	2.6, 2.7 (2.6)	2.3, 2.8 (2.5)	2.6, 2.8 (2.7)
Did you feel like interacting fast?	2.8, 2.9 (2.8)	2.8, 2.9 (2.8)	2.4, 2.6 (2.5)	2.7, 2.9 (2.8)

0 = Fully disagree, 3 = fully agree.

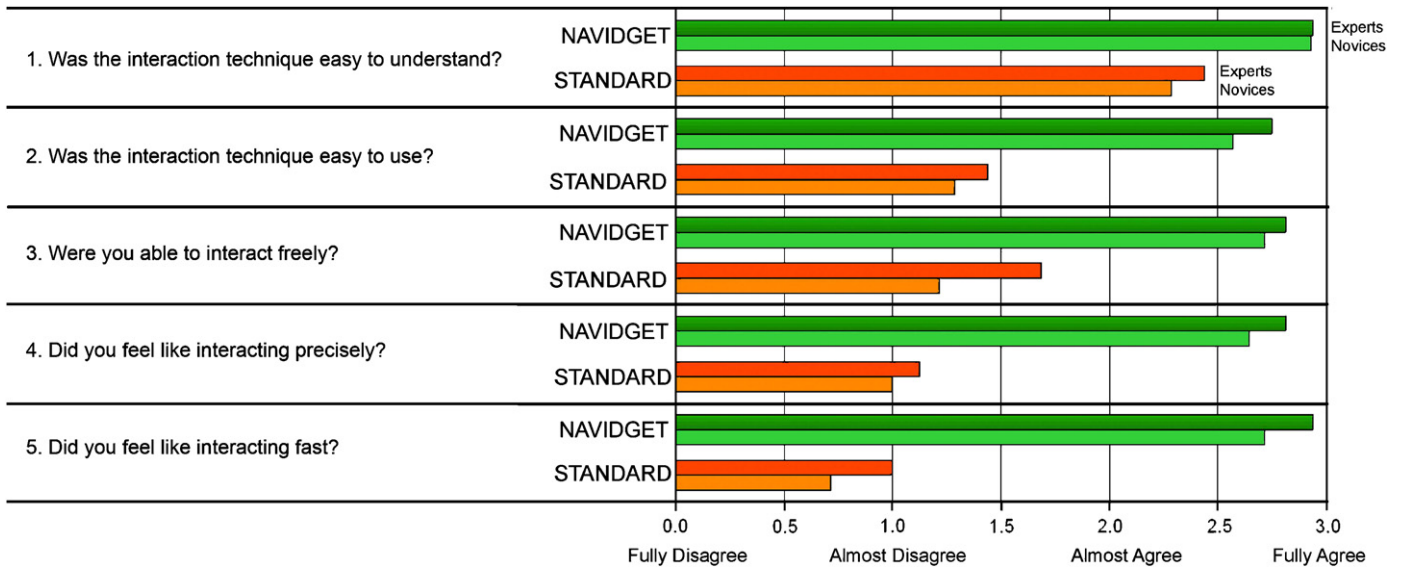


Fig. 11. General satisfaction for Navidget and standard controls.

the experiments, the subjects were exactly knowing what was going to happen. They were able to easily control where they wanted to move. Thanks to smooth camera movements, no motion sickness has occurred.

Zooming + orbiting is the standard technique that relates the most to Navidget. We have noticed that this technique may perturb the user when large scenes were used. Indeed, the whole scenes move around the user, which can be disturbing. Moreover, many occlusions can occur. This makes this technique little adapted to dense environments. Of course, *Zooming+orbiting* is well suited for exocentric tasks where single objects are observed. On the other hand, Navidget seems better suited when users navigate large 3D scenes with egocentric viewpoints. Navidget seems better suited when true real-time cannot be insured, too. Further investigations have to be conducted to confirm this.

Of course, this study does not mean that Navidget is better than the standard controls in general. It simply shows that, for camera positioning tasks, Navidget can be more appropriate. The current version of Navidget is adapted to target-based navigation, where the destination is visible from the current viewpoint; it does not fit well with naive exploration of 3D scenes. During the experiments we conducted, some participants complained that they were not able to return to previous camera positions.

We advised them to use another standard control (e.g. fly and look-around) as an alternative. But they did not want to take back the direct control of the camera movements. Consequently, we have extended the initial Navidget as described below.




5. Extended Navidget

In order to allow the users to navigate using only Navidget, we introduced two new sets of gestures. The first one allows them to come back to previous states while the second one makes it possible to turn the camera in any direction.

5.1. Back-gesture

A back-gesture (see Table 2) entails repositioning the camera in its previous state. In other words, the camera follows the last trajectory path that has been played, but in the reverse direction. From the user's perspective, this action can be seen as an "undo" action. Technically, all the new positions and orientations of the camera are pushed in a stack. An undo gesture entails popping the stack to move the camera back to its previous state.

Table 2
Extended Navidget gestures

Gesture	Resulting action from gestures
	Focus on an area
	Go back to previous location
	Turn the view

Navidget, associated with this new functionality, allows the users to explore 3D environments in a new manner. This type of 3D exploration can be related to website navigation, where users explore new areas of interest by way of hyperlinks, and return to previous states using “back” actions. We think that such simple web-based navigation can help the users in their navigation tasks since they benefit from a well-established mental model.

For example, let’s consider a user who wants to observe several objects that are simultaneously visible in a 3D scene. With a standard approach, the user would have to reach the first target, then would have to search for the second one from the new location, and so on. This can be difficult, particularly because wayfinding in 3D environments is a difficult task. With the extended version of Navidget, the user can focus on the first object, then can come back to the initial location by way of a simple stroke on the screen, and then focus on a new target, and so on. This is similar to what users do when navigating through web pages.

5.2. Turn-gestures

Another feature we have implemented to extend the initial version of Navidget is the possibility for users to turn the view around their current location. This is done by way of vertical and horizontal strokes (see Table 2). Following the Navidget philosophy, the users do not control directly the movements of the camera. By drawing a vertical (resp. horizontal) stroke, users indicate the system that they want to look up/down (resp. left/right). At the end of the stroke, the camera smoothly turns in the sketched direction.

Technically, a 45° horizontal/vertical rotation is applied to the camera. With a standard 90° camera *field-of-view*, this entails moving the viewing vector towards one of the edges of the screen. Hence, the users can look everywhere around them using simple strokes.

5.3. Summary

The gestures for the extended Navidget are summarized in Table 2. The number of gestures that the users have to remember is small. Consequently, this extended version of Navidget does not require an additional learning period.

We have not conducted a formal study to evaluate these new controls. However, according to our first user feedback, it appears that this extended version of Navidget works well. The users have not experienced any difficulties in assimilating the new Navidget functionality, which provide more control for navigation in 3D environments.

6. Further uses of the Navidget approach

In the previous sections, we have described Navidget in terms of camera positioning tasks. In the sections that follow we will present several implementations that show how the Navidget approach can be useful in many different situations.

6.1. Defining a 3D vector

Navidget makes it possible to orient a 3D vector around a target area, from 2D input interaction. For camera positioning tasks, this vector refers to the viewing direction at which the user wants to visualize a target. This 3D vector can also be used for other tasks. We present two implementations in the following sections.

6.1.1. 3D pointing

With current 3D applications, pointing generally refers to the selection of a 3D point using a picking operation from the standard 2-DOF pointing devices. By using the Navidget approach, users are no longer limited to 3D positions; they can also control orientations as well as distances to targets. This can be beneficial in numerous situations. For example, an adapted design of the Navidget widget can be useful for the definition of force vectors in interactive physical simulation. could be envisioned, such as spot light positioning, virtual drilling, clipping plane positioning, and so on. Fig. 12 illustrates an example of 3D pointing.

6.1.2. Rotation of objects

The Navidget approach can also be useful for manipulation tasks. It allows users to define 3D axes of rotation, around which the object can spin. Current interfaces for object rotations are either based on a virtual trackball or canonical decomposition of the rotation space. In the first case, many rotations are difficult or even impossible to achieve (e.g. rotation around the viewing direction axis). In the second case, the users have to compose the rotation they want to achieve using iterative adjustments. On the other hand, with the Navidget approach, a rotation axis can be defined very quickly and easily. The benefits of this approach for such manipulation tasks are similar to those

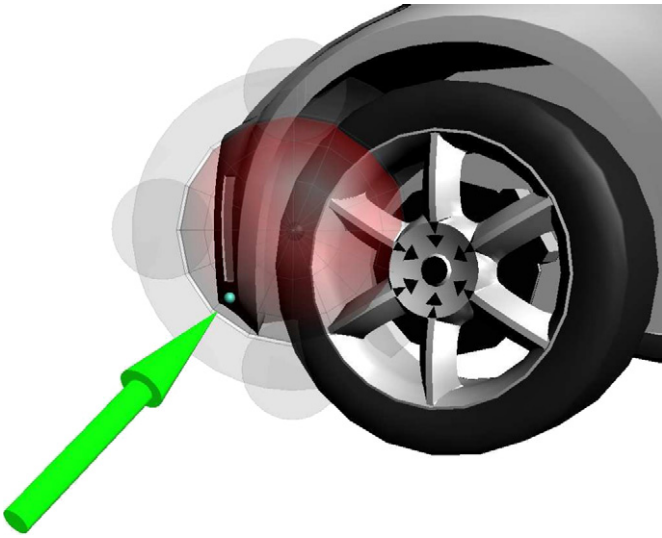


Fig. 12. The Navidget approach for 3D pointing.

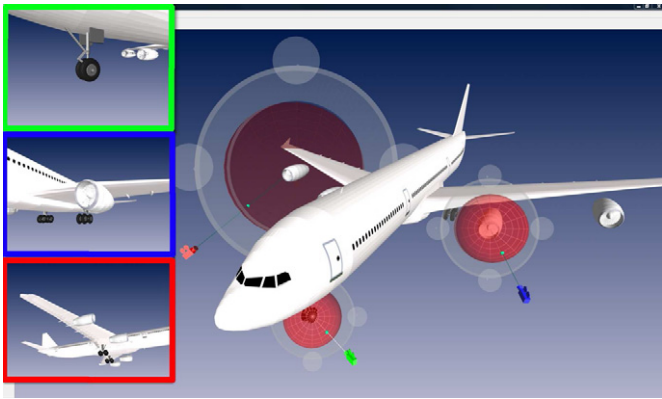


Fig. 13. Multiple Navidget for simultaneous inspection.

for camera positioning tasks: users can concentrate on what they want to do, rather than concentrating on how they can do it.

6.2. Multiple cameras

Until now, we have considered the use of one Navidget at a time. Below we shall discuss some implementations where it could be beneficial to use several Navidget interfaces at the same time. In these examples, the user's viewpoint does not move automatically towards the targets. The widgets stay visible in the scene and the user can access each of them.

6.2.1. Defining several control viewpoints

It can be beneficial to visualize a 3D scene from several viewpoints, at the same time. If we consider the example of 3D physical simulation, it may be necessary to see several parts of the scene simultaneously in order to control the evolution of physical phenomena. To this end, we developed a small prototype, using several widgets associated with corresponding preview windows that can

be positioned in the scene, as illustrated in Fig. 13. During the simulation, if required, the user can easily adjust the virtual cameras.

6.2.2. Animation

Similarly, the use of multiple widgets can be useful for animation purposes. Indeed, animation paths are specified as a sequence of key (view) points with orientation defined by positioning several widgets into the scene. Next, the system associates time values and automatically calculates continuous intermediate values using an interpolation method.

Navidget naturally fits well with the definition of the key viewpoints, as the interface has been designed to allow fast and easy camera positioning around focus areas. In the test application we developed, the user can define key viewpoints by using several widgets and then play a camera motion along an interpolated path. We use uniform cubic B-Splines to obtain a smooth animation by interpolating the acquired key viewpoints. The positions and orientations of the virtual key cameras can be adjusted at any time.

6.3. Collaboration

We think that collaborative applications can benefit from Navidget, too. Two examples are illustrated below, which we implemented as prototypes.

6.3.1. Shared displays

In front of large screens, several participants can visualize the same image. This is of great benefit for collaborative work. On the other hand, the viewpoint is shared by all participants, so if one of them wants to focus on a specific area, the whole group is affected by the applied camera motions. We propose a new approach where several participants can share global visualization of a scene (e.g. a car) while simultaneously focusing on different specific parts of it (e.g. the tyres, the lights, and so on). To achieve this, the participants must be equipped with personal pointing devices (e.g. lasers, Wiimotes, and PDAs). Then, all of them control their own Navidget on the large screen. This allows them to reach private views in the scene. These private views can be displayed on the side of the main window as illustrated in Fig. 14. They can also be sent to the participants' personal devices (e.g. PDA and TabletPCs).

6.3.2. Distant collaboration

We have previously described the use of the Navidget approach for 3D pointing. This functionality is particularly beneficial for distant collaborative work. Indeed, distant collaborators have to understand one another despite the fact they cannot see each other. Consequently, fast and easy 3D pointing from 2D inputs is very beneficial for this type of applications. Indeed, the collaborators can highlight precisely what they are talking about.

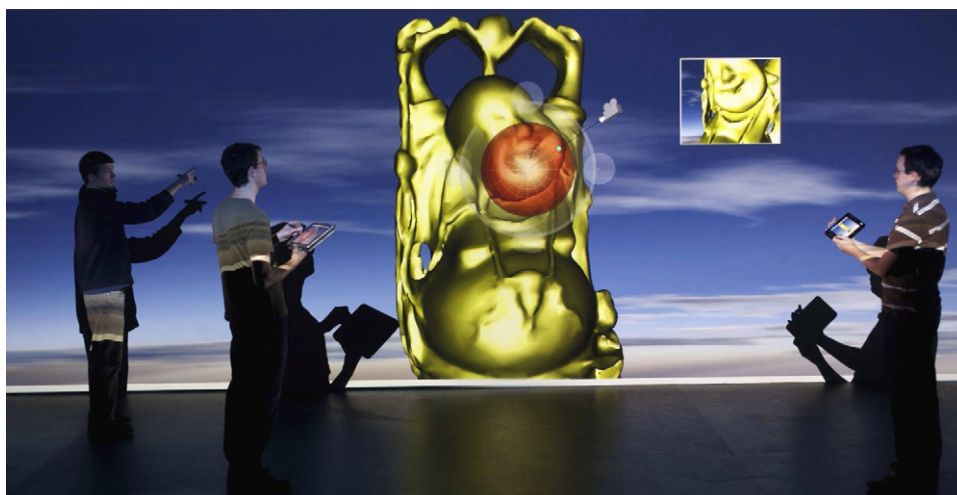


Fig. 14. The shared view is displayed on the large screen while Navidget allows the participants to get personal views on their TabletPCs. ©CNRS Phototheque/C. Lebedinsky.

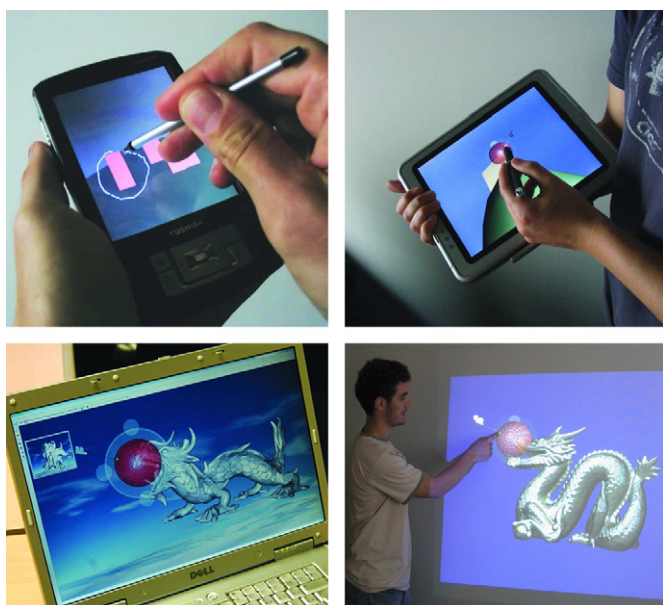


Fig. 15. Navidget on various systems.

7. Conclusion

In this paper, we have proposed a new approach for camera positioning tasks. The key idea of this technique is to provide good controls and feedbacks to the user, in order to allow them to easily position the camera around focus areas.

We have implemented Navidget with OpenGL, OpenSG, and OpenGL|ES. The technique is based on mouse events. It can be used with a wide variety of platforms: PDAs, TabletPCs, standard workstations, large touch screens, and so on, since the entire technique can be controlled by means of a simple 2D gesture sequence. Fig. 15 shows some examples. Moreover, the same technique can be used with immersive virtual environments, by means of a virtual ray controlled by some trackers for example. This portability is

one of the strengths of Navidget. Users do not have to learn new context-dependent techniques when dealing with new interactive systems. Instead, they benefit from a unified technique. This enhances user mobility.

The user study has shown that the usability of Navidget was good. It has also shown that, Navidget can be a good alternative to the standard controls that have been available for years on many 3D viewers. With these controls, many basic navigation tasks are difficult to perform. Navidget can make the completion of these tasks easier.

In addition to camera positioning tasks, we have shown some examples where the Navidget approach can help the user in many interaction tasks. This new interface changes the way we interact with 3D data.

Acknowledgments

This work has been supported by the French research agency ANR. We would like to thank Wolfgang Stuerzlinger for his fruitful comments.

References

- Bier, E.A., 1987. Skitters and jacks: interactive 3d positioning tools. In: SI3D '86: Proceedings of the 1986 Workshop on Interactive 3D Graphics. ACM, New York, NY, USA, pp. 183–196.
- Blender. (<http://www.blender3d.com>).
- Burtnyk, N., Khan, A., Fitzmaurice, G., Balakrishnan, R., Kurtenbach, G., 2002. Stylecam: interactive stylized 3D navigation using integrated spatial & temporal controls. In: Proceedings of UIST '02. ACM Press, New York, pp. 101–110.
- Buxton, W.A.S., 1995. Chunking and phrasing and the design of human–computer dialogues. pp. 494–499. Available at (<http://portal.acm.org/citation.cfm?id=212970#>).
- Chen, M., Mountford, S.J., Sellen, A., 1988. A study in interactive 3-d rotation using 2-d control devices. In: Proceedings of SIGGRAPH '88. ACM Press, NY, pp. 121–129.
- Forsberg, A.S., Dieterich, M., Zeleznik, R.C., 1998. The music notepad. In: Proceedings of UIST'98, pp. 203–210.

- Grosjean, J., Coquillart, S., 1999. The magic mirror: metaphor for assisting the exploration of virtual worlds. In: Proceedings of 15th Spring Conference on Computer Graphics, pp. 125–129.
- Hachet, M., Dècle, F., Guitton, P., 2006. Z-goto for efficient navigation in 3D environments from discrete inputs. In: Proceedings of VRST, pp. 236–239.
- Hachet, M., Dècle, F., Knödel, S., Guitton, P., 2008. Navidget for easy 3D camera positioning from 2D inputs. In: Proceedings of IEEE 3DUI—Symposium on 3D User Interfaces, pp. 83–89.
- Hanson, A.J., Wernert, E.A., 1997. Constrained 3D navigation with 2d controllers. In: Proceedings of VIS '97. IEEE Computer Society Press, Silver Spring, MD, pp. 175–182.
- Igarashi, T., Kadobayashi, R., Mase, K., Tanaka, H., 1998. Path drawing for 3D walkthrough. In: Proceedings of UIST'98, pp. 173–174.
- Igarashi, T., Matsuoka, S., Tanaka, H., 1999. Teddy: a sketching interface for 3D freeform design. In: Proceedings of SIGGRAPH '99. ACM Press, Addison-Wesley, New York, Reading, MA, pp. 409–416.
- Khan, A., Komalo, B., Stam, J., Fitzmaurice, G., Kurtenbach, G., 2005. Hovercam: interactive 3D navigation for proximal object inspection. In: Proceedings of SI3D '05. ACM Press, New York, pp. 73–80.
- Mackinlay, J.D., Card, S.K., Robertson, G.G., 1990. Rapid controlled movement through a virtual 3D workspace. In: Proceedings of SIGGRAPH '90. ACM Press, New York, pp. 171–176.
- Mackinlay, J.D., Robertson, G.G., Card, S.K., 1994. Moving viewpoint with respect to a target in three-dimensional workspace. US Patent 5,276,785.
- Schmalstieg, D., Encarnação, L.M., Szalavári, Z., 1999. Using transparent props for interaction with the virtual table. In: Proceedings of SI3D '99. ACM Press, New York, pp. 147–153.
- Singh, K., Grimm, C., Sudarsanam, N., 2004. The ibar: a perspective-based camera widget. In: Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology. ACM Press, New York, NY, USA, pp. 95–98.
- Strauss, P.S., Carey, R., 1992. An object-oriented 3d graphics toolkit. In: SIGGRAPH '92: Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques. ACM, New York, NY, USA, pp. 341–349.
- Tan, D.S., Robertson, G.G., Czerwinski, M., 2001. Exploring 3D navigation: combining speed-coupled flying with orbiting. In: Proceedings of CHI '01. ACM Press, New York, pp. 418–425.
- Thorne, M., Burke, D., van de Panne, M., 2004. Motion doodles: an interface for sketching character motion. *ACM Transactions on Graphics* 23 (3), 424–431.
- Ware, C., Osborne, S., 1990. Exploration and virtual camera control in virtual three dimensional environments. In: Proceedings of SI3D '90. ACM Press, New York, pp. 175–183.
- Zelevnik, R.C., Forsberg, A.S., 1999. Unicam—2d gestural camera controls for 3D environments. In: Proceedings of SI3D'99, pp. 169–173.
- Zelevnik, R.C., Herndon, K.P., Hughes, J.F., 1996. Sketch: An interface for sketching 3d scenes. In: Proceedings of SIGGRAPH 96, Computer Graphics Proceedings, Annual Conference Series, pp. 163–170.
- Zelevnik, R.C., LaViola, Jr. J.J., Acevedo, Feliz, D., Keefe, D.F., 2002. Pop through button devices for VE navigation and interaction. In: VR '02: Proceedings of the IEEE Virtual Reality Conference. p. 127.

DRILE: an immersive environment for hierarchical live-looping

Florent Berthaut
University of Bordeaux -
SCRIME - LaBRI
berthaut@labri.fr

Myriam
Desainte-Catherine
University of Bordeaux -
SCRIME - LaBRI
myriam@labri.fr

Martin Hachet
INRIA - LaBRI
hachet@labri.fr

ABSTRACT

We present Drile, a multiprocess immersive instrument built upon the *hierarchical live-looping* technique and aimed at musical performance. This technique consists in creating musical trees whose nodes are composed of sound effects applied to a musical content. In the leaves, this content is a one-shot sound, whereas in higher-level nodes this content is composed of live-recorded sequences of parameters of the children nodes. Drile allows musicians to interact efficiently with these trees in an immersive environment. Nodes are represented as *worms*, which are 3D audiovisual objects. Worms can be manipulated using 3D interaction techniques, and several operations can be applied to the live-looping trees. The environment is composed of several virtual rooms, i.e. group of trees, corresponding to specific sounds and effects. Learning Drile is progressive since the musical control complexity varies according to the levels in live-looping trees. Thus beginners may have limited control over only root worms while still obtaining musically interesting results. Advanced users may modify the trees and manipulate each of the worms.

Keywords

Drile, immersive instrument, hierarchical live-looping, 3D interaction

1. INTRODUCTION

In the past decade, the live-looping technique has become more and more used in musical performances, either for solo singers who create their accompaniment or for instrumental or electronic improvisations. However, it has some limitations due to the use of hardware controllers, such as the impossibility of creating and manipulating complex musical structures or the impossibility of modifying all the parameters of recorded loops. We believe that 3D immersive environments can support the evolution of this technique into a more advanced system for musical performances. Indeed these environments provide interesting possibilities for 3D interaction with multiprocess instruments, i.e instruments that are composed of several sound synthesis processes. They also enable new way of building, visualizing and navigating complex musical structures as 3D shapes. Immersion, e.g using large stereoscopic displays, may be as valuable for the audience as it is for musicians. For example, comprehension of what musicians are doing and involvement in the musical performance may be improved by the use

of stereoscopic vision and large screens. Finally, these immersive environments are often used in other fields for collaborative tasks, and thus they may be appropriate for collaborative musical performance.

In section 3, we describe an evolution of the live-looping technique, called *hierarchical live-looping* technique. Then, in section 4, we present *Drile*, our immersive musical instrument which relies on this technique. As it can be seen on figure 1, the musician wears 6DOF tracked stereoscopic glasses, and he interacts with the instrument, displayed on a large screen, using the *Piivert* input device. We define the components of our instrument in section 4.2, then we describe how we represent and interact with the 3D *live-looping trees* in section 4.3. Finally, we present the live-looping scenes in section 4.4 and the collaboration/learning possibilities brought by our approach in section 4.5.

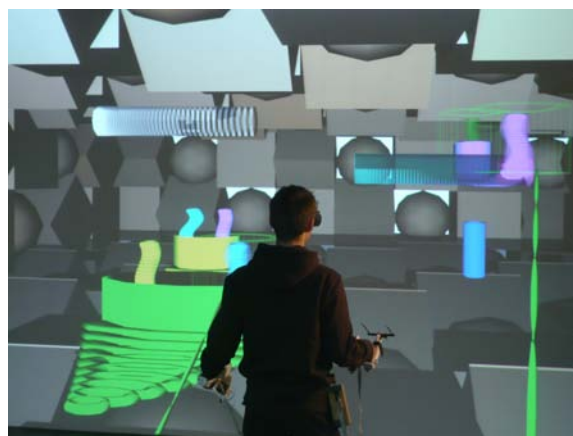


Figure 1: Drile used by one expert musician. The live-looping scene contains a three-level tree on bottom left, a two-level tree on top right, 1 leaf node/worm and 2 tunnels (scattering/reverb and color hue/pitch).

2. RELATED WORK

2.1 3D Virtual Instruments

Many existing immersive instruments focus on navigation in musical environments, like the virtual groove in the Phase project [10] or the audiovisual grains in Plumage [3]. These applications allow musicians to play precomposed musical structures, but they do not give access to the structure of the synthesis processes.

Other immersive instruments are single process instruments, i.e. instruments that allow to interact with only one synthesis process, such as the Virtual Xylophone, the Virtual Membrane, or the Virtual Air Guitar developed by Mäki-Patola et al. [4] and the sculpting instruments developed by Mulder [6]. The application devel-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

NIME2010, June 15-18, 2010, Sydney, Australia
Copyright 2010, Copyright remains with the author(s).

oped by Mike Wozniowski et al. [12] relies on users movements to either control the spatialization of pre-recorded sound sources, or apply effects on the sound of an acoustic instrument. All these applications aim at being ordinary instruments, with more control over synthesis processes than navigation tools, that one may use for musical performance. However, they do not take advantage of the possibilities brought by 3D environments in terms of multiple processes handling and structures creation.

An interesting application designed by Polfreman [8] takes advantage of 3D environment to build complex musical scores including hierarchical organisation of elements. Nevertheless it is not designed for musical performance but rather for composition.

2.2 Live-Looping

Live-Looping is a musical technique that consists in recording musical loops from an audio or control (e.g MIDI messages) input and stacking these loops to quickly build musical structures or sound textures. It has several advantages over other electronic playing techniques such as triggering sequenced loops, or applying effects to different tracks of a prepared song. First of all, there is usually no temporal quantization so it enables more natural musical patterns as opposed to midi sequences. It may fit any musical genre since any musical input and any time-signature can be used. Finally, one can rely on prepared loops or improvise every part of a musical structure. It is mostly used for live performances by different instrumentalists such as guitar players, beat-boxers, or electronic musicians¹. It is also an interesting tool for music writing, allowing quick sketching of songs.

Live-Looping originates from tape-delay systems and was explored by contemporary composers such as Terry Riley or Steve Reich. This looping system was then implemented in hardware effects racks or guitar pedals as the Gibson Echoplex, Digitech Jamman, Boss RC20 and so on. Usual operations include record, play and stop/mute. One can also overdub, i.e add material to an existing loop, multiply the length of a loop, and reverse a loop. Live-looping has evolved with digital loopers, such as Sooperlooper² or Freewheeling³, which include new possibilities such as synchronization, timestretching, graphical interfaces and so on.

Most live-looping systems deal with audio input, so that the loops are audio buffers. However, some of them, such as LiveLoop⁴, deal with control inputs, such as MIDI or OpenSoundControl messages or other events. They record and playback series of events, which may be either sound triggers or synthesis/effects parameters controls. This control live-looping has many advantages over audio live-looping. First of all, loops can be easily modified since each event is separable. This enables timestretching and addition/removal of events. Notes and effects controls may be recorded independently, so that they can be modified separately. Finally the same recorded loops can easily be rerouted on different synthesis processes.

Some novel instruments are based on live-looping. For example, the BeatBugs [11] are small input devices that record rhythms played by users and repeat them. Recorded rhythms can then be modified and several devices can be synchronized, enabling collaborative musical interaction. Another very interesting application is Fijuu [7], in which users manipulate 3D audiovisual shapes using a gamepad. Each shape is associated with a specific sound synthesis process which is triggered when the shape is distorted. Audiovisual effects can also be applied on produced sounds and audio loops can be recorded for each shape. These loops can then be accelerated, muted and amplified. However, creation and modification of musical structures remain limited, as well as the inter-

action possibilities.

3. HIERARCHICAL LIVE-LOOPING

Basic live-looping does not enable the creation and manipulation of complex musical structures, but only lists of recorded sequences. It also reduces the possibilities of advanced modification of recorded loops, especially when the content is audio data. We propose to expand the advantages of control live-looping with what we call *hierarchical live-looping*. The tree structure used in *hierarchical live-looping* is based on the study of Marczak [5]. This section focuses on the concept of hierarchical live-looping while section 4 describes its implementation as a 3D immersive instrument.

3.1 Principle

We define *live-looping trees* containing leaves $l : < c_l, x >$ and nodes $n : < ch, c, x >$. x is a list of audio effects. c_l is a one-shot, i.e no sequences or loops, musical content which can be a soundfile or another synthesis process. ch is a list of children nodes. c is a musical content composed of live-recorded sequences of children events. These events are effects parameters and triggers of the musical content. The audio effects available for all nodes are pitch, volume, distortion and reverb. In addition to these audio effects, the tempo of sequences can be modified, and other high-level musical effects could be added. Each node and leaf also has a content play mode, which can be *trigger*, i.e the content is played until its end, *normal*, i.e the content is played until its end or the reception of a stop event, and *loop*, i.e the content is looped until it receives a stop event. The loop mode is not used for leaves,

Control and audio data follow different paths in *hierarchical live-looping*. Control data go top-down the tree because nodes contain their children sequences. Audio data go bottom-up the tree, which means that the sequences of sounds contained in the leaves will be successively modified by the effects of their parent nodes. The audio and control data flows and the structure of a *live-looping trees* can be seen on figure 2.

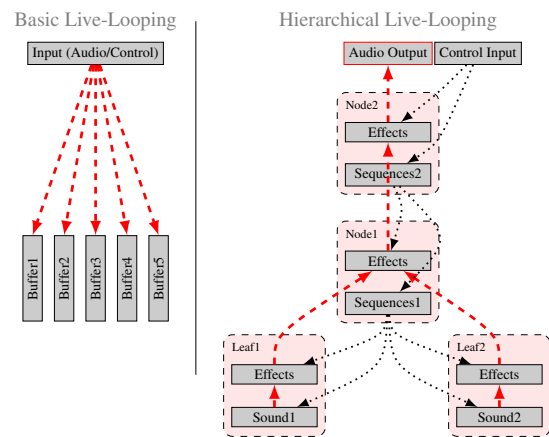


Figure 2: Basic live-looping data flow (left) and hierarchical live-looping control (in black) and audio (in red) data flows.

Hierarchical live-looping has several advantages over traditional live-looping approaches. First of all, it allows musicians to build musical sequences of any complexity by creating and manipulating trees. A simple example can be seen in figure 3.

It also provides direct access to any level of the trees. One may trigger only the snare or kick to produce a fill, or the drums node to completely modify the rhythm. Effects can be applied to any node independently, whether they are leaves or high-level nodes, in which case the effects will affect all their children. For example one may filter all the drums by modifying the drums node, or

¹<http://www.livelooping.org/>

²<http://www.essej.net/sooperlooper/>

³<http://freewheeling.sourceforge.net/>

⁴<http://code.google.com/p/liveloop/>

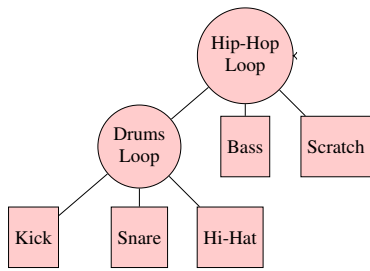


Figure 3: Hip-Hop live-looping tree example.

pitchshift only the hi-hat.

Live-looping trees also adapt to user with different levels of experience. Beginners can play with tree roots, thus triggering musically interesting parts and applying simple effects to it. Advanced users can go down the tree and manipulate each node. They may change any sequence by re-recording it, or modify the effects associated to the different nodes.

Furthermore, advanced users can easily collaborate to build musical structures. For example, they may add nodes to existing trees, or duplicate nodes to remix loops prepared by others.

Finally, several live-looping trees can coexist, allowing musicians to experiment with non-synchronized loops. This may be useful to create complex sonic textures.

3.2 Node operations

Two operations are available for the nodes. Nodes content can be triggered, and nodes audio effects can be modulated. Triggering a leaf node will play its sound whereas triggering a higher-level node will play its sequences of events and parameters. Audio effects parameters can also be modified. In leaves, this will directly affect the musical content, i.e the audio file, whereas in higher-level nodes it will affect the resulting audio output of all children. Musical effects, such as tempo modification, affect higher-level nodes sequences.

3.3 Tree operations

We propose a set of operations for the live-looping trees, which are shown in figure 4: *build*, *merge*, *duplicate*, *extract*.

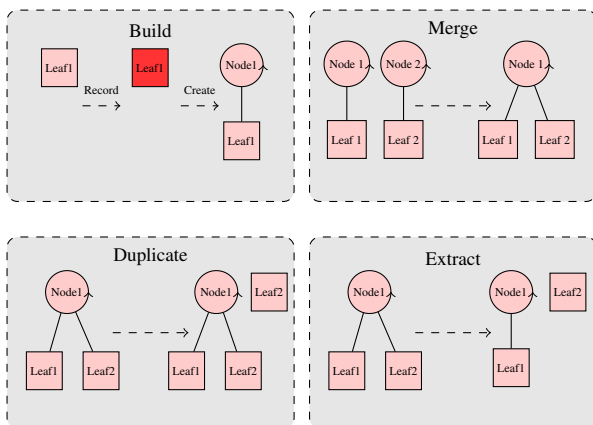


Figure 4: Hierarchical live-looping operations.

The *build* operation has two steps. The first step starts the recording of the events of a node. The second step stops the recording, creates a parent node and define the recorded sequences of events as its content. The playing mode of the parent content is automatically set to *loop* so that its children sequences keep playing, as it would occur with a traditional live-looping application. The play-

ing mode can then be changed to *trigger* or *normal*. When a build operation is performed on a node that already has a parent, this operation simply records sequences over existing ones.

The *merge* operation consists in linking two trees by merging two parent nodes into a single parent node. The oldest sequence of these nodes, i.e the one which was recorded first, is taken as a pulse. Another sequence could be chosen as the pulse, but the usual playing technique in live-looping is to first record the pulse, then the other loops. All the other sequences are then synchronized to multiples of this pulse according to their length, as depicted in figure 5.

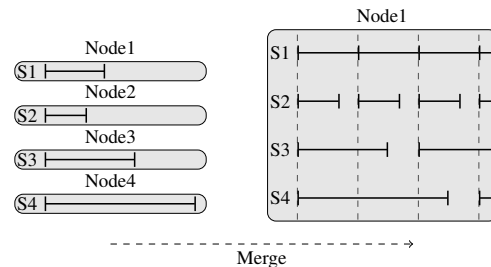


Figure 5: Example of sequences synchronization when merging several nodes. Here the oldest sequence is S1.

Thus the parent node synchronizes all its children together, which allows musicians to build regular rhythms. The *build* and *merge* operations can be combined if multiple sequences are recorded at the same time. In that case, all simultaneously built parent nodes are merged into a single node. One must note that this operation is not possible if both nodes have parents, since it may lead to too much changes in the live-looping tree.

The *duplicate* operation consists in duplicating a part of a live-looping tree. This is a useful operation because hierarchical live-looping results in nodes being locked to their parents content, as one node can not have two parents, for musical structure understanding reasons. Thus nodes can not be used in two different sequences at the same time. Using the *duplicate* operation, one can copy a node in order to use it to build another sequence, with different effects parameters for example.

With the *extract* operation, one may remove any node and its children from a live-looping tree, making it the root of a new tree, or a leaf if it has no children.

3.4 Implementation

Hierarchical live-looping adds new theoretical possibilities to basic live-looping. However, its relevance heavily depends on its implementation. Indeed operations on trees and manipulations of nodes need to be efficient. 3D immersive environments provide new possibilities in terms of interaction, visualization and immersion. They are thus fitted to the control of complex musical structures, especially in a performance context.

4. DRILE

Drile implements the *hierarchical live-looping* technique in a 3D immersive environment. In this implementation, leaves contain multisampled sounds which can be played normally or using granular synthesis. Live-looping trees are represented by 3D virtual structures. Musicians interact with these structures in front of a large screen. They are immersed in the 3D environment thanks to stereoscopic glasses and head-tracking. A video of Drile can be watched on <http://vimeo.com/9206485>.

4.1 Technical setup

Drile is composed of three applications which run on Gnu/Linux: *drile-audio*, *drile-ui* and *drile-ui-display*. *Drile-audio* handles the

sound processes and it outputs audio via the Jack sound server. It also uses LV2 audio effects plugins, and VAMP audio analysis plugins. Drile-ui handles the scene graph of the 3D environment. They may run on two separate computers and communicate via OpenSoundControl messages. The stereoscopic display is done by two instances of drile-ui-display synchronized with drile-ui and connected to separated projectors, one for each eye. In addition, one user (or several with split screens) can be equipped with tracked glasses, to improve immersion and interaction with the environment. Users wear passive stereoscopic glasses and the sound is played using an external usb soundcard and two active speakers.

4.2 Worms, Tunnels and Piivert

In Drile, nodes of live-looping trees are represented by what we call *worms*. These worms, which can be seen on figure 1, have several graphical parameters such as color hue, size, transparency, scattering and so on. These parameters are associated to the nodes audio effects parameters, so that modifying the worms appearance modifies the nodes audio effects. Current mappings are size/volume, color hue/pitch, transparency/distortion and scattering/reverb. They were chosen as a result of a user study, following user preferences and trying to preserve the independance of graphical perceptual dimensions. One must note that the mappings are relative, not absolute. For example, two worms with the same color hue do not necessarily have the same pitch, but their initial pitch is modified by the same amount. Graphical parameters thus give the current value of the effect, like a cursor on a graphical slider, which is essential for efficient interaction.

Worms shapes are associated to the analysed spectrum of nodes audio outputs, to allow users to identify which worm is associated to a node. Finally, each worm rotation on the y-axis is associated to the position of the read pointer in its associated node content. This allows musicians to follow the playback of nodes sequences.

Worms graphical parameters, and thus their associated sound parameters, can be modified by grabbing and sliding the worms through what we call *tunnels*. Two of them can be seen on figure 1. Each tunnel is associated to one or several graphical parameters, and one or several scales for each parameter. The tunnels are made of a succession of thin cylinders which reflect the values of the parameters scales. The scales can be continuous or discrete. For example, if one associates size with pitch, one may define musical notes with an array of sizes. One may modify only one worm at a time by passing it through a tunnel, but one may also grab and move a tunnel to modify several worms at the same time, for example to mute several loops.

To interact with the environment, we use an input device called Piivert [1]. Piivert is a bimanual input device that combines 6DOF tracking, using infrared targets detection, with high-sensitivity pressure sensing. It allows musicians to benefit from the possibilities of graphical interaction while preserving accurate musical controls. Force-resistive sensors are positionned under the thumb, index, middle and ring fingers, allowing us to detect different gestures. These include low-level gestures such as hit and pressure, and high-level percussion gestures such as flam, three-strikes roll and four-strikes roll. This device allows us to select and grab worms and tunnels using a ray-casting technique, i.e a virtual ray that sticks out of the device and goes through the 3D environment. To grab a worm or tunnel, users perform a pressure gesture with their thumb. Low-level gestures transmit vibrations to the worms with the virtual ray, and thus trigger the associated node. Hit gestures play the entire content of the node while pressure gestures play the content using random small grains of the content, which results in granular synthesis for the leaves. Tunnels scale presets can also be changed by selecting the tunnels and performing a hit gesture. Finally, high-level gestures are used to trigger hierarchical live-looping operations. A user holding this device can be seen in figure 1.

4.3 Hierarchy

Common 3D selection and manipulation techniques such as the virtual ray technique are more efficient for objects at short distances, as evaluated by Poupyrev et al. [9]. Furthermore, as the size parameter is used to control the volume of worms, allowing continuous modification of worms depth may disrupt the perception of worms size. Thus we propose to keep most worms and tunnels on what we call the *interaction plane*, at a fixed distance. This preserves spatial and temporal accuracy for musical actions such as selecting and manipulating worms and tunnels. Depth and navigation in the environment may then be used for actions that require less temporal accuracy.

4.3.1 Live-looping trees

As depicted in figure 6, live-looping trees are represented by connected worms organized by depth. Each level of the tree is displayed at a discrete depth. The level that is the closest to the users is on the *interaction plane*. Worms children are placed in kind of appendages. Musicians may only select and grab the worms on the *interaction plane* and their direct children. By physically moving in front of the screen, musicians can easily point directly at children with the virtual ray, thanks to head-tracking. When grabbed, the children jump to the depth of the *interaction plane* so that they can be modified using the *tunnels*. To access lower-level and higher-level worms, musicians respectively pull and push one of the worms, as depicted in figure 6. When going down the tree, high-level worms move towards users and disappear. Their appendage remain visible, to indicate that displayed worms are children. When users grab and move the root worm of a tree, all the tree follows its translations. When users grab a children worm, move it and release it, it jumps back to its parent appendage.

The immersive environment allows us to display the live-looping hierarchy without overloading the interface and disrupting the interaction. Indeed, manipulation of nodes is done within the *interaction plane* whereas access to the trees levels relies on the depth of the environment and on the use of head-tracking. These simultaneous manipulations of trees and nodes would be more complicated with a 2D interface, since they could not be done in separated dimensions.

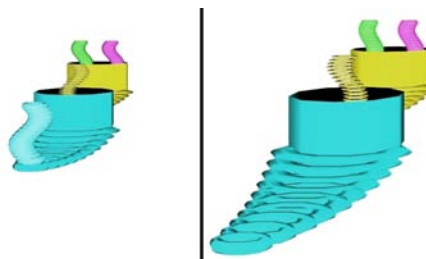


Figure 6: Three-levels live-looping tree (left). Pulling a worm goes down the tree (right).

4.3.2 Tree Operations

The *build* operation needs to integrate closely in a sequence of musical gestures since it records a musical loop. Thus we have defined a high-level gesture with Piivert, actually a ring-middle finger flam. First the user needs to select a worm with the virtual ray. When the record step is triggered, the worm changes shape to highlight its rotation around the y-axis. When the create step is triggered, the created parent node appears and the child worm moves behind it, in its appendage, as depicted on figure 7.

The *merge* operation is a less temporally critical operation. This is done by colliding worms. When two worms collide for more than 1 second, the system will try to merge them. All children of one of the worms are transferred to the other, and this empty

node is deleted from the scene. Children worms are automatically arranged behind their parent worm in its appendage, as depicted in figure 7.

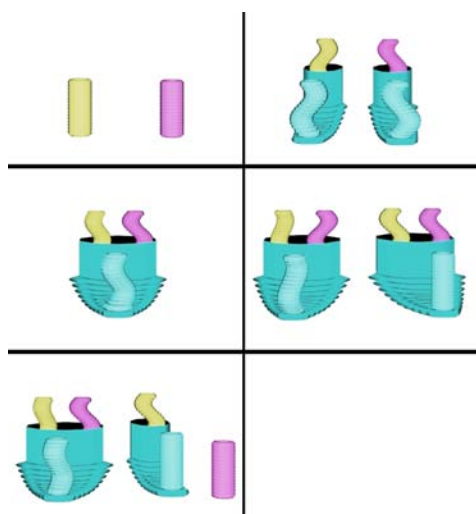


Figure 7: Live-looping trees operations (from right to left, top to bottom): Build, Merge, Duplicate, Extract

The *duplicate* operation is also done graphically by grabbing a worm with two rays, i.e two hands, and stretching it, as it can be seen on figure 7. A copy of the stretched worm and all these children, with all contents, is then added to the scene. The stretched worm thus become the root of a new live-looping tree, and is brought to the *interaction plane*.

Since the *extract* operation is the opposite of the build operation, it is also triggered by a high-level Piivert gesture. This gesture is a flam with the same fingers but in the opposite direction, i.e middle finger before ring finger. The extracted worm is brought to the *interaction plane*, its children remaining behind it, as depicted in figure 7. This results in a new live-looping tree.

4.4 Live-looping scenes

As explained in section 2.2, the live-looping technique often includes the idea of *scenes*. A scene is composed of several loops which are synchronized, or which simply fit together musically. With hardware systems, like guitar pedals, scenes are selected using buttons or rotary encoders. Thus each scene is only associated with a number. With 2D looping software, like Freewheeling, scenes are selected with a menu, and are associated with a name. Their musical features, such as tempo or mood, may thus be better described.

In Drile, we define scenes as 3D rooms containing several worms, which can be leaves or high-level nodes of pre-built live-looping trees, and several tunnels.

When the user walks backward from the screen and reaches a specific distance, the camera moves backward, revealing the grid of scenes, as depicted in figure 8. A scene can then be selected simply by looking at it. The head movement needs to be a little exaggerated though, as the selection is computed from the position and orientation of the user's head. Finally, the user move to the selected scene simply by walking towards the screen.

Such an approach has several advantages. Visual properties of each scene/room, such as colors, 3D objects, walls and so on, may be used to reflect the musical "mood" of the live-looping trees. Users can refer to these properties to choose the musical content they want to play with. The appearance of the scenes is defined in a configuration file with a XML syntax.

Different tunnels can be chosen for each scene, as audio effects needs may be different for different musical content.

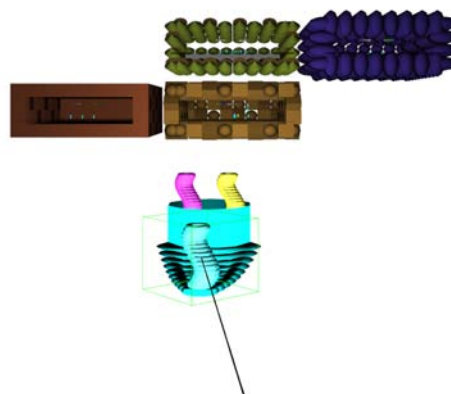


Figure 8: View of all scenes (here 4). Here one tree is being moved from one scene to another.

A worm can be brought from a previous scene, to make smooth transitions between musical parts. This worm can be a leaf, with a single sound, so that musicians may play it a last time while starting the new musical part. It may also be the root of complex tree, so that the new musical part will consist in modifying the previous tree by progressively adding new sounds.

Performances may follow very different paths, as any transition between scenes is possible and may be done at anytime.

Finally, each scene has a floor. Musicians can bypass worms or complete trees by moving them below this semi-transparent floor. Their audio data is then sent another stereo Jack output, which can be connected to earphones for example. This may be used as a monitor system to prepare trees and then play them, or it can be simply used to quickly mute trees.

4.5 Learning and Collaboration

Thanks to *hierarchical live-looping*, learning Drile is progressive. Indeed, following Birnbaum et al. classification of new instruments in dimension spaces [2], Drile happens to be between two different configurations, as depicted in figure 9. New users may begin by interacting only with the root worms of pre-built live-looping trees. This allows them to manipulate musically interesting sequences without much expertise. To push this idea further, we have developed an interaction technique in addition to the ones available with Piivert, which we call the *bucket*. Users manipulate a virtual bucket using a 6DOF tracked device with a single button. They may then grab and move only worms on the *interaction plane*. They are not allowed to trigger the sequences, but only pass these worms through tunnels, or bypass them. Interaction is quite simple, i.e grabbing and moving worms, and the musical possibilities are reduced to high-level processes manipulation. Thus required expertise, musical control and degrees of freedom are low.

When users get accustomed to this subset of Drile interaction techniques, they can switch to Piivert. By going down the tree, musicians progressively access rawer musical content, i.e simpler sequences and finally single sounds. Symmetrically, the degree of interaction needed to produce an interesting result, i.e the required expertise, rises. At first, they may interact only with the root worms, this time being able to trigger the musical content. When gaining expertise, they may access lower levels of pre-built live-looping trees, to trigger their nodes and make more complex manipulations. Finally, after more learning, they may modify trees structures by using the available operations, and even build new trees. Obviously required expertised increases as hierarchical live-looping operations and low-level worms manipulations are complex. The number of degrees of freedom also increases because Piivert enables new musical gestures and 3D interaction techniques.

Symmetrically, the musical control evolves from high-level control of a musical process to a complete control of the sound and musical parameters of each worm.

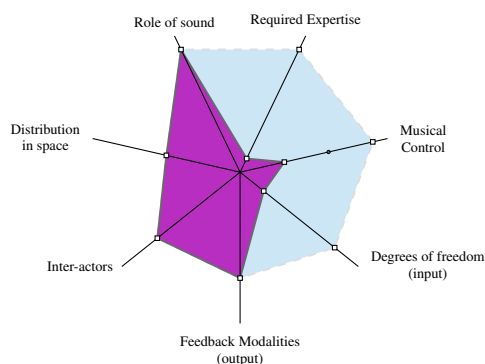


Figure 9: Drile Dimension Space, in purple for beginners with buckets and only access to root worms, in blue for advanced users with piivert and access to all worms of the live-looping trees.

Collaborative performances may rely on these different expertise levels. An advanced user can easily collaborate with one or several beginner users, as seen in figure 10. In this configuration, the expert builds live-looping trees, and pass them to the beginners so that they can manipulate worms at the foreground, i.e interesting musical sequences. As the bucket does not allow to manipulate the tunnels, the expert may also choose which tunnels should be used by the beginners.



Figure 10: Collaboration between one advanced user with Piivert (left) and two beginner users with buckets

5. CONCLUSION

In this paper, we presented Drile, an immersive multiprocess instrument especially suited for musical performances. It relies on the concept of hierarchical live-looping, which enhances basic live-looping by allowing musicians to create and manipulate complex musical structures. Drile enables efficient use of this technique thanks to the possibilities brought by immersive environments. In particular, musicians make use of head-tracking, navigation in 3D environments and combination of hierarchy visualization with 3D interaction. Moreover, Drile is well adapted to learning and collaboration between users with different levels, because live-looping trees include different steps of musical interaction complexity.

Collaboration between several beginners users and a single advanced user is simple to setup with one screen, as beginners do

not need head-tracking to use the *bucket* technique. Collaboration between advanced users, i.e using virtual rays and head-tracking, on the same screen could be implemented with a more complex hardware setup. 3D avatars may also be used for distant collaboration, as multiple environments could be synchronized using OpenSoundControl messages together with OpenGL internal synchronization features. Following Birnbaum et al. classification, distribution in space would then be maximized. Another interesting issue is the immersion of the audience. They may obviously be equipped with stereoscopic glasses, but one can imagine various stage setups. Musicians and the audience may both face the same screen, though musicians may occlude parts of the screen. 3D avatars may also be used to display musicians while they interact in front of another screen. Future work will investigate these different issues.

6. REFERENCES

- [1] F. Berthaut, M. Hachet, and M. Desainte-Catherine. Piivert: Percussion-based interaction for immersive virtual environments. In *Proceedings of the IEEE Symposium on 3D User Interfaces*, 2010.
- [2] D. Birnbaum, R. Fiebrink, J. Malloch, and M. M. Wanderley. Towards a dimension space for musical devices. In *Proceedings of the 2005 International Conference on New Interfaces for Musical Expression (NIME05), Vancouver, BC, Canada*, pages 192–195, 2005.
- [3] C. Jacquemin, R. Ajaj, R. Cahen, Y. Ollivier, and D. Schwarz. Plumage: Design d’une interface 3d pour le parcours d’échantillons sonores granularisés. In *Proceedings of the Conférence Francophone sur l’Interaction Homme-Machine(IHM’07)*, 2007.
- [4] T. Mäki-Patola, J. Laitinen, A. Kanerva, and T. Takala. Experiments with virtual reality instruments. In *Proceedings of the 2005 International Conference on New Interfaces for Musical Expression (NIME05), Vancouver, BC, Canada*, 2005.
- [5] R. Marczak. Etude d’une représentation hiérarchique liant micro et macro-structures musicales. Master’s thesis, University of Bordeaux, 2007.
- [6] A. G. Mulder. *Design of virtual three-dimensional instruments for sound control*. PhD thesis, Simon Fraser University, Canada, 1998.
- [7] J. Olive and S. Pickles. Fijuu: <http://www.fijuu.com/>.
- [8] R. Polfreman. Frameworks 3d: composition in the third dimension. pages 226–229, Pittsburgh, USA, Carnegie Mellon University, 2009.
- [9] I. Poupyrev, S. Weghorst, M. Billinghurst, and T. Ichikawa. *Egocentric object manipulation in virtual environments: Empirical evaluation of interaction techniques*. 1998.
- [10] X. Rodet, F. Gosselin, P. Mobuchon, J.-P. Lambert, R. Cahen, T. Gaudy, and F. Guedy. Study of haptic and visual interaction for sound and music control in the phase project. In *Proceedings of the 2005 International Conference on New Interfaces for Musical Expression (NIME05), Vancouver, BC, Canada*, 2005.
- [11] G. Weinberg, R. Aimi, and K. Jennings. The beatbug network: a rhythmic system for interdependent group collaboration. In *NIME ’02: Proceedings of the 2002 conference on New interfaces for musical expression*, pages 1–6, Singapore, 2002. National University of Singapore.
- [12] M. Wozniowski, Z. Settel, and J. Cooperstock. A spatial interface for audio and music production. In *Proceedings of the International Conference on Digital Audio Effects (DAFx), 2006*, 2006.

Contents

Curriculum Vitae	vii
Introduction	1
1 Mobile 3DUI	7
1.1 Introduction	7
1.2 Interaction from keystrokes	8
1.2.1 Jump and Refine	8
1.2.2 Z-Goto	10
1.3 Interaction from the mobile device touch-screens	11
1.3.1 Improving stylus interaction with elastic feedback	11
1.3.2 Sketchball	13
1.4 New inputs for mobile devices	14
1.4.1 Tangimap for bi-manual interaction	15
1.4.2 3 DOFs elastic controller	16
1.5 Conclusion	18
2 Touch-based Interaction	21
2.1 Introduction	21
2.2 Navidget	22
2.2.1 The technique	22
2.2.2 Usages	23
2.3 Towards multi-touch interaction with 3D data	24
2.3.1 Seemingly good ideas	24
2.3.2 Extending 2D RST to 3D objects	26
2.4 Conclusion	28

3	Immersive Interfaces	31
3.1	Introduction	31
3.2	Input devices	31
3.2.1	The CAT	31
3.2.2	PIIVERT	32
3.2.3	Opportunistic music	33
3.3	Image-Sound-Human interaction	34
3.3.1	Audio-visual mapping	34
3.3.2	Interacting with the sound	35
3.3.3	A new virtual instrument for immersive performances	36
3.4	Conclusion	38
4	Perspectives and Conclusion	39
4.1	Future work	39
4.1.1	Touch-based 3DUI, again.	39
4.1.2	Creation of 3D content for non-expert users	40
4.1.3	Usage-guided interaction and immersive interaction for everyone.	41
4.2	Conclusion	42
	Bibliography	44
	Appendix	51