Deliverable D4.3/4.4

Production Tools for Electronic Arenas: Event Management and Content Production

ABSTRACT

This document comprises a combined deliverable made up of Deliverables D4.3 and D4.4 to the eRENA project. This document considers how the production of events in electronic arenas, might be supported. The concern is with the 'behind-the-scenes' activities which are necessary in staging an event and how those activities might be best supported technically. Basing our research on the practical experience accumulating in eRENA, and on the field studies of production work performed, we present novel technologies and orientations to the support of production process, these principally focus on technologies for supporting events as they occur in real-time and include: proposals and demonstrations of virtual cameras for capturing activity in electronic arenas, techniques for sonifying the activity of participants in an electronic arena to give an audible sense of the status of events, a physical environment containing tangible interfaces to production software to facilitate timely direction and production work, and a system for mapping data from participants to enable its flexible interpretation in media-rich environments. It also begins to consider how content for events might be developed both off-line as well as improvised in real-time. The document contains reports of public artistic events which have driven the developments here as well as stimulated novel cross-project collaborations. Throughout there is a concern to reflect critically on the status of the technologies and concepts explored, where possible through formal analysis, and how they might influence and learn from other developments in the eRENA project.

Document eRENA – D4.3/4.4
Type Deliverable report with video material
Status Final
Version 1.0
Date September, 1999

Authors Sabine Hirtes, Michael Hoch, Bernd Lintermann and Sally Jane Norman
Zentrum für Kunst und Medientechnologie (ZKM)
John Bowers, Kai-Mikael Jää-Aro, Sten-Olof Hellström and Malin Carlzon
Royal Institute of Technology (KTH)

Task D4.3/D4.4

Table of Contents

Preface

Document Overview....7
The Production Requirements of Electronic Arenas....7
Evaluation and Reflection....8
Structure of this Document....10
Relationship of this Document to the eRENA Workplan....11

Chapter One

Exploring Requirements for Event Design and Management in Electronic Arenas: The *Real Gestures, Virtual Environments* Extended Performance Workshop....13
The *Real Gestures, Virtual Environments* eRENA Workshop....13
Technology-free performance experimentation
at the IIM: playing with constraints....14
Technologically-extended performance experimentation at the ZKM:
scrambling screen-stage boundaries....17

1.2. Technologically Generated Theatrical Doubles....19
1.2.1. Multiple screens to multiply avatars: *Ubu Roi*....19
1.2.2. Single screen to multiply avatars: *Shadow Puppets*....21
1.2.3. Low-tech pointers to theatrically effective high-tech: *Monte Carlo*....22

1.3. Specific Experimentation with the mTrack Vision-based Interface....24
1.3.1. Tracking coloured regions....25
1.3.2. Motion detection....25

1.4. Real-time Animation and Motion Blending....28

1.5. Specific Experimentation with xfrog and a Magnetic Interface....30
1.5.1. Metaphorical versus direct mapping: testing performer ease....30
1.5.2. Streaming live data into live performance....33
1.5.3. Control data extraction, control value abstraction....34
1.5.4. Making data dance: (semi-) autonomous mapping switches....36
Conclusions....37

APPENDIX....39

Chapter Two

A General Framework for Transforming, Mapping and Choreographing User Interface Data for Performance Purposes in Electronic Arenas 40
2.1. Introduction 40

2.2. Motivation 40

2.3. Goals 41

2.4. Data Pipeline 42

2.5. System Discussion 43
2.5.1 Usability 43

2.5.2 Flexibility 43
2.5.3 Extensibility 44

2.6. System Architecture Description 44
2.6.1 Network 44
2.6.1.1 Evaluation Strategy 45
2.6.2 Plugin Architecture 46
2.6.3 Software Architecture 46
2.6.3.1 Network Layer 46
2.6.3.2 Manipulation Layer 47
2.6.3.3 User Interface Layer 47

2.7. Implementation 47
2.7.1 Attributes 47
2.7.2 Nodes 48
2.7.3 User Interface 49

2.7.3.1 Network 49
2.7.3.2 Sub-Networks 51

2.7.3.3 Usability Features 52

2.8. Summary 53

Appendix A 54

Appendix B 56

Chapter Three

Event Management in Electronic Arenas by Visualising Participant Activity and Supporting Virtual Camera Deployment 58
3.1 Introduction 58

3.2. Critical Examination of Existing Research into Camera Deployment and Control in Virtual Environments 60
3.2.1. Supporting Camera Deployment 60
3.2.1.1. The Virtual Cinematographer 60
3.2.1.2. Automatically Generated Illustrations 61
3.2.1.3. Variably Parameterisable Cameras 61
3.2.2. Supporting View Manipulation 62
3.2.2.1. Manipulation Metaphors 62
3.2.2.2. Simple Algorithmic View Control 62
3.2.2.3. More Sophisticated Algorithmic View Control: Procedures and Constraints 63
3.2.2.4. Automating Viewpoint Placement 64
3.2.2.5. Through-the-Lens-Camera-Control 65
3.2.3. Conclusions 65
3.2.3.1. Real-time Operation 65
3.2.3.2. Mass Social Participation 66
3.2.3.3. Understanding Rules of Practice 66
3.2.3.4. Hybrid Interaction Methods in a Working Division of Labour 66
3.2.3.5. Scripted and Improvised Action 67
3.2.3.6. Geometry, Physical Movement and Capturing Action 67

3.3. Activity-Oriented Camera Deployment and Control 67
3.3.1. Activity Heuristics 68
3.3.2. The 'Spatial Model' of Awareness 69
3.3.3. Mapping Activity and Awareness 70

3.4. Algorithms 71
3.4.1. Algorithmically Deploying Cameras: Identifying Groups 71
3.4.2. Algorithmically Deploying Cameras: Heuristics for Initial Shots 71
3.4.2.1. Centre of Gravity 72
3.4.2.2. Centre of Viewpoint 72
3.4.2.3. Bisecting View Directions 73
3.4.3. Activity and Awareness Maps as Giving Dynamic Potentials to Autonomous Cameras 73

3.5. Implementation 74
3.5.1. Interpreting the Spatial Model in SVEA 75
3.5.2. Computing and Displaying Activity Maps in SVEA 75
3.5.3. Camera Deployment and Real-Time Interaction with Activity Maps in SVEA 77

3.6. Future Work 78

Chapter Four

Supporting Event Management by Sonifying Participant Activity 80
4.1. Introduction.....80
4.1.1 The Research Field of Sonification 80
4.1.2. Sonifying Participant Activity in Electronic Arenas 81

4.2. What Is To Be Sonified?.....83

4.3. Criteria for the Design of Sound Models.....84

4.4. Implementation.....85
4.4.1. A Pulse and Voice Simulation.....86
4.4.2. A Techno Musical Sound Model.....87
4.4.3. A Granular Synthesis Sound Mode 88

4.5. Experimental Evaluation 88
4.5.1. Experimental Design 88
4.5.2. Data Analysis.....90

4.6. Conclusions.....95

4.7. Future Work.....96

Chapter Five

Round Table:
A Physical Interface for Virtual Camera Deployment in Electronic Arenas 99
5.1. Introduction.....99

5.2. Mixed Reality / Shared Environments´ 101

5.3. Round Table with Interaction Blocks.....101

5.4. Vision Based Tracking.....103

5.5. Interaction.....103

5.6. Co-ordinating Multiple Virtual Cameras.....105

5.7. Updating the Relationships Between Visualisation,
Physical Interaction and Virtual Cameras 106
5.8. Current Status, Conclusions and Future Work 107

Chapter Six

***Blink:* Exploring and Generating Content for Electronic Arenas 109**
6.1. Introduction.....109

6.2. World Construction and Camera Control Interfaces 112

6.3. *Blink* Image Gallery.....116

6.4. Implementation, Installation and Performance Details 124

6.5. Experience in Performance.....125

6.6. Conclusions and Future Work 127

6.7. Acknowledgements.....128
References.....129

Deliverable D4.3/4.4
Production Tools for Electronic Arenas:
Event Management and Content Production

Preface

John Bowers
Royal Institute of Technology (KTH), Stockholm, Sweden

Document Overview

This document comprises a combined deliverable made up of Deliverables D4.3 and D4.4 to the eRENA project of the i3 schema of the ESPRIT-IV research action of the European Communities. eRENA is concerned with the development of electronic arenas for culture, art, performance and entertainment in which the general citizen of the European Community might actively participate supported by advanced information technology. Within this general context, this document considers how the production of events in electronic arenas, the topic of Workpackage 4, might be supported. The concern here is with the 'behind-the-scenes' activities which are necessary in staging an event and how, when appropriate, those activities might be best supported technically. Basing our research on the wealth of practical experience accumulating in eRENA, and on the dedicated field studies of production work performed in the project, we present a number of novel technologies and orientations to the support of the production process. In the current document, these principally focus on technologies for supporting events as they occur in real-time and include: proposals and demonstrations of virtual cameras for capturing activity in electronic arenas, techniques for sonifying the activity of participants in an electronic arena to give an audible sense of the status of events, a room-sized physical environment containing tangible interfaces to production software to facilitate timely direction and production work, and a system for mapping data from participants to enable its flexible interpretation in media-rich environments. It is also begins to consider how content for events might be developed both off-line as well as improvised in real-time. The document starts and finishes with reports of public artistic events which have driven the developments here as well as stimulated novel cross-project collaborations. Throughout there is a concern to reflect critically on the status of the technologies and concepts explored, where possible through formal analysis, and how they might influence and learn from other developments in the eRENA project.

0.1. The Production Requirements of Electronic Arenas

In the Periodic Progress Report (PPR) to the project, we describe how eRENA has developed in Year 2 a characteristic concept of 'electronic arena' to guide its explorations. *An electronic arena deploys mixed reality technologies to create environments for potentially large-scale real-time participation in media-rich cultural events.* The five key-terms (mixed reality, large-scale participation, real-time, media-richness, and cultural events) all give our research agenda a specificity. For example, we are concerned with virtual reality technologies very centrally but not

just VR. Importantly, we are concerned with the web of practices and physical realities into which VR systems are inserted, the mixture of the virtual and physical.

We are concerned with large-scale real-time participation. This emphasis has extensive influence on the work reported in this document. A major consequence is that many production solutions developed for theatre, television or film are not readily workable in the electronic arena context. For example, as noted in a number of chapters in this deliverable, computer animation in the film industry, though highly developed technically and with considerable penetration into public discourse, is very much geared to off-line solutions computed with the aid of a detailed script. Real-time events (especially when 'live' and even more so when improvised or involving public participation) require solutions here and now, and in response to developments which may well be unscripted.

We are concerned with media-rich environments - environments where, at the extreme, multiple sources of content can come together, be processed or otherwise combined in real-time, and be distributed or transmitted to multiple destinations. This raises challenges beyond many more familiar technologies for managing multimedia content, especially as our concern is also to deepen the participation of the general citizen in popular media and artistic works to the point of being (ultimately) co-creators of content.

The work we report in this document, and in related deliverables, offers some practically grounded technologies for the support of the production of events in electronic arenas. All the technologies discussed in this document are geared for real-time, interactive, 'live' operation and performance. Some were developed specifically to support performer interaction in real-time settings in creating or working with media-rich materials. Others are targeted at production and direction staff working to ensure the right resources are available at the right time for an event to continue or for experience of it to remain engaging. In each case, our developments are motivated by practical experience of working on ambitious demonstrators or in conducting challenging workshops under the aegis of eRENA.

In this regard, cross-workpackage collaborations and influences is a notable feature of this document, and, as such testifies to our concern to respond to the recommendations of eRENA's first year reviewers that such collaborations should be initiated more strongly. Two partners in Workpackage 4 (KTH and Nottingham) were major contributors to Deliverable D7a.1 which reported the eRENA project's first demonstrator in inhabited television. Nottingham developed event management and virtual camera control technologies directly out of their work in Workpackage 4 that they deployed in the demonstrator. For their part, KTH conducted a detailed ethnographic study of the production work involved in the inhabited television demonstrator and brought that to bear on shaping their research agenda in Workpackage 4. This has lead to the development of novel ideas for virtual camera control and real-time event management which complement those developed at Nottingham to make for a rich set of tools available for demonstrators in Year 3 of the project.

The ZKM and GMD, the other partners who have worked in Workpackage 4, have further overlapping collaborations with KTH and Nottingham in other areas of the project. Again this has ensured that a strong influence from practical experience in project demonstrators (cf. Deliverable D7b.1) and on performances delivered to Workpackage 6 (cf. Deliverable 6.2) has had the chance to influence work in this workpackage. Again, this has been brought about through actual collaborations between partners, through, for example, KTH's ethnographic study

of the performance that forms the focus of Deliverable D6.2. This entails the strong practical grounding of the technical work that appears here.

This practical experience has lead us to emphasise the real-time aspects of event support more than we expected would be the case at the outset of Year 2. This is not to say that off-line, pre-production or post-production work is now irrelevant to Workpackage 4. On the contrary, it remains firmly on our agenda for Year 3. Indeed, the relationship between live action and content produced for broadcast is likely to emerge as central when we have the opportunity in Workpackage 4 to more deeply analyse our experience working on the second set of Workpackage 7 demonstrators (presented in Deliverables D7a.2 and D7b.2).

Another implication of our reflections on the practical work required to realise events in nascent electronic arenas is that it is hard to keep issues of event management and design on the one hand separate from audience participation and content production on the other. Indeed, an improvised event with public participation precisely runs together all of these matters in its realisation. For this and other reasons (see Section 0.5 below), we have decided to deliver D4.3 and D4.4 together as a combined document.

0.2. Evaluation and Reflection

Essential to our approach in Workpackage 4 has been rigorous reflection on our efforts. Each of the chapters in this document which report work shown to the public has endeavoured to analyse what works and what does not under the stresses of public performance. When possible (and it is not always), this has involved actual audience consultation through post-event discussions. The workshop format followed by the ZKM enabled important technologies of the project developed at the ZKM to trialed by demanding artists wishing to creatively yet expressively work with new technology. The sonfication research reported by KTH has proven itself amenable to preliminary experimental test to investigate whether the sonic differences that the investigators propose to be audible in fact are. In addition, several of the technologies developed within this workpackage but delivered elsewhere (e.g. the Nottingham camera control and event management technologies) have been the subject of ethnographic appraisal of their usefulness in work-related settings. In these respects, Workpackage 4 has been concerned to respond strongly to the recommendations of the Year 1 review of eRENA that a critically reflective and, where possible, explicitly evaluative (e.g. through formal tests or social scientifically oriented study) emphasis should be developed. It is hoped that all the technical developments documented here are seen to be well motivated in the light of our experience and, when it has been possible, tested in public settings or in some other way which will prompt critical reflection.

It must be emphasised (and we say this again in Chapter 5) that the status of much of the work is that the pathways from study through requirements to implementation can be traced. However, the user testing which would initiate a second iteration in design is yet to start. This is the case for the physical interfaces developed in Chapter 5, for example. There though we do still state a program of study for the assessment of what we have done, such evaluations being imminent early in Year 3.

0.3. Structure of this Document

This document is structured as follows. After this Preface, there follow six chapters. Chapter 1 describes a two week long workshop organised by the ZKM in collaboration with the International Puppetry Institute. This workshop sought to explore new performance practices which might be developed to exploit interactive technologies in novel ways while avoiding some of the theatrical pit-falls documented in earlier work by partners in eRENA (see especially the critical account of staging a virtual reality opera in Deliverable D2.3 from the first year of the project). A reading of the chapter should demonstrate how useful it was to base a workshop around puppetry, rather than live human action or for that matter around choreographed avatars, as this allowed numerous explorations of questions of scale and of the relationships of the real to the virtual. The chapter closes with some substantive recommendations for staging events in electronic arenas which subsequent work in the workpackage (see especially the connections between Chapters 1 and 6) has been attentive to.

A critical issue in the staging of performance events in electronic arenas is identified in Chapter 1 and this has strongly motivated the technical design work reported from the ZKM in Chapter 2. This concerns the difficulties that exist in creating a flow of events when numerous reconfigurations of software (e.g. an animation system) and hardware (e.g. sensor endowed peripherals) have to take place between episodes within a production. Accordingly, a layer which supports the mapping, transformation and calibration input data before communicating with animation or sound processing devices is recognised as coherently separable. Chapter 2 presents the overall specification for such a software layer and describes its current status of development. In Workpackage 4, we anticipate that this will be a major development and provide an important resource to productions in electronic arenas as a sophisticated yet easy to use graphical programming environment which could elegantly coordinate real-time performances. The further development of this software, as described at the end of Chapter 2, is likely to be of cross-partner interest (especially to KTH with their experience of identifying the same problems and their history of working with similar, yet more specific, software solutions) and cross-workpackage significance (especially to Workpackage 6 with its concern for interaction technologies).

Chapters 3, 4 and 5 present a trajectory of work which responds to KTH's identification of requirements for event support technology in Deliverable D7a.1. These chapters establish the centrality of 'virtual cameras' to the production of events in electronic arenas. A virtual camera, as a source of image-content sourced from an electronic arena, needs to be appropriately deployed and controlled if it is to capture (and not miss) the action. Chapter 3 develops an argument that a great part of the existing work on virtual cameras and the management of virtual events assumes that what is of interest is non-real-time film-like applications, commonly with just a small number of participants as actors, well-scripted and with highly conventional ways of editing from one camera to another being followed. In this way, a great deal of the existing literature (including quite well-known and commonly cited work) misses the challenges raised by electronic arenas. Chapter 3 then goes on to develop an alternative perspective (referred to as 'activity oriented camera deployment and control') whereby various indices of participant activity are used to give production staff clues as to where the action might be in a large scale electronic arena and how best to deploy cameras to capture it. An application for visualising activity in electronic arenas is presented, along with various algorithms for the deployment of cameras and the animation of their paths in ways which are consonant with the concept of activity oriented control and deployment. The notion of activity measures as a resource for

controlling viewpoint in an electronic arena is returned to in Deliverable D6.3, where individual and group navigation issues are examined.

Chapter 4 presents an argument that production staff to an event in an electronic arena might benefit from the presence of auditory displays to cue them to significant changes in the activity or location of participants in a large scale environment. The literature on data sonification is reviewed and some sound models which sonify real-time data concerning participants is proposed to complement the visualisation work presented in Chapter 3. The design criteria for this sound model are carefully stated and one model in particular is identified, described in some detail, and subjected to psychophysical testing. The results of formal analysis indicate that subjects in the study are able to understand the dimensions of variation in the model on the basis of a verbal description of it and make reliable judgements as to which dimension is varying when sound samples are played to them. On closer inspection of the data, several features become apparent which give guidance as to how the model could be improved before it is subjected to more work-like evaluation.

Chapter 5 argues that, to support the timeliness of action on the part of production personnel that live events require, physical interfaces must be developed so support interaction with many of the applications to be worked on in support of an event. Accordingly, the ZKM and KTH have collaborated in developing a room-sized environment containing carefully designed projection surfaces and physical icons (or 'phicons') to mediate a user's production work. As a focus for this development, the visualisation and camera deployment application of Chapter 3 is further developed so that interaction with it can take place using phicons in addition to the conventional GUI and mouse-driven interaction of earlier versions. The work of Chapter 5 gives us the fullest indication of how we envisage the electronic arena 'production suite' of the future, at least with respect to how to support camera deployment in an effective way.

Chapter 6 closes the deliverable with a description of a public event which formed part of the Nottingham Now98 arts festival. This event, a live performance involving the improvised construction of virtual forms and environments, develops earlier work conducted by KTH in Year 1 of eRENA. As Chapter 6 makes clear, the further refinement of the technologies involved was made in direct response to the experience of the earlier versions in public performance. In addition, KTH's collaboration with the inhabited television demonstrators suggested a model of multi-camera editing in virtual environments which lead to new technological developments. What is more, some of the main recommendations arising from the puppetry workshop described in Chapter 1 were observed in the design of the environment in which the performance was realised. Chapter 6 demonstrates a work in which content creation for electronic arenas comes to the forefront and how such processes can be interactively accomplished in real-time. On the basis of analysis of performer experience working with the software developed for the work, together with audience feedback, Chapter 6 closes with suggestions for future work which converge with the developments described in Chapters 2 and 5, as well as indicating strong interconnections with the event management concepts worked with by Nottingham in this workpackage yet reported in Deliverable D7a.1.

0.5. Relationship of this Document to the eRENA Workplan

This document combines Deliverables D4.3 and D4.4. These deliverables are the output at the end of Year 2 of Workpackage 4, Tasks 4.3 and 4.4 respectively. These tasks run through to the

end of the project, where a combined deliverable, D4.5, is planned for the end of Year 3. We decided to adopt the combined format this year as well for a number of positive reasons. First, as noted above, the issues in the two tasks overlap conceptually just as their time-frames do. Especially as we have begun to sharpen our focus on a distinctive electronic arena concept, we have found this to be more so. The most radical challenges to existing literatures and hence the most opportunity for innovative work comes from seeing 'event design and management' (the topic of Task 4.3) and 'audience participation and content production' (the topic of Task 4.4) as intertwined processes. One thing we hope to have shown in this deliverable (and in those other areas of the project where its influence can be felt, e.g. Deliverable D6.3) is that production tools can be put in the hands of participants, at least in some version. Participants (performers, inhabitants, whatever) have to manage their views on an electronic arena just as directors and camera operators may have to manage cameras. It seems appropriate, then, to consider the applicability of similar paradigms in both cases. Thus, while we discuss the sonification work in Chapter 4 primarily as giving a production resource, this gives us strong ideas for how world sound might be generated as content available to participants. Much of our work has this duality: a resource for production can also be a resource for participation. And, in the extreme case of real-time large-scale audience participation events, production and participation fold into one another.

Another reason, and a related one, for combining Deliverables D4.3 and D4.4 is that to separate out those parts of, say, the puppetry workshop or the performance described in Chapter 6 which were related to event design technologies and those parts related to content creation would be artificial and lead to an atomised presentation of coherent work.

While we prefer this merged presentation in a longer deliverable, the reader should not get the impression that we have covered all the issues we intend to in Tasks 4.3 and 4.4 already. Our coverage of the research issues as outlined in the revised Project Programme document is not yet complete. We have not yet considered support for story boarding. We give some consideration to the design of virtual architecture in Chapter 6 but in a rather idiosyncratic way. The support of more conventional stagings for action needs to be considered. We have not in this workpackage considered how best to support audience participation, though in eRENA there has been progress on such matters reported in Deliverable D7a.1 and D6.3. We have only begun to consider how to manage sound as content for electronic arenas.

These omissions set our agenda for Year 3, along with the refinements and testing our current technologies as outlined in the next six chapters. Important to the unfolding of this work will be deeper consideration of the Workpackage 7 demonstrators which emerged in the latter half of Year 2. Much of the work here has built upon needs recognised consequent upon the demonstrators which delivered in Month 18. What is reported here is now available for future demonstrators. Reciprocally, Workpackage 4 will profit from the recently reported demonstrators to the end of Year 2, some of which are still under development. We hope that the text which follows, especially in its combined format, is able to demonstrate the profitable interplay between demonstrators and technical development forms the heart of eRENA's cycling between Workpackage 7 on the one hand and Workpackages 4, 5 and 6. In the current case, we believe that structuring the work in this way has levered the enhanced collaborative and cross-workpackage relations deemed essential by Year 1 reviewers.

Work has been performed in Workpackage 4 by KTH, Nottingham, ZKM and GMD. This document was edited by John Bowers (KTH) based on contributions from KTH and ZKM.

Chapter One
Exploring Requirements for Event Design and Management in Electronic Arenas: The *Real Gestures, Virtual Environments* Extended Performance Workshop

Sabine Hirtes, Michael Hoch, Bernd Lintermann and Sally Jane Norman
Zentrum für Kunst und Medientechnologie (ZKM), Karlsruhe, Germany

John Bowers
Royal Institute of Technology (KTH), Stockholm, Sweden

1.1 The *Real Gestures, Virtual Environments* eRENA Workshop

The August 1998 eRENA performance workshop was a two-week event, held for the first week at the International Institute of Puppetry (IIM) in Charleville-Mezieres (France) and for the second week in the Medientheater of the Zentrum für Kunst und Medientechnologie (ZKM), Karlsruhe, where it terminated with a Saturday evening public demonstration of workshop activities, presented as "work in progress" rather than as finished spectacle. Entitled *Real Gestures, Virtual Environments*, this workshop sought to explore the artistic ramifications of performance visibly amplified in real time, by using motion capture and camera tracking technologies to pilot computer graphics (CG) during an actual staged event. We investigated various types of mapping to be applied with theatrical coherence and effectiveness, in a performance situation involving human actors, physical objects imbued with life (puppets), and computer-generated actors driven by human actors, i.e. entities imbued with life in keeping with theatrical principles analogous to those governing puppetry (Norman, 1996a).

Upstream of the August event, a technical core group from the ZKM Institute for Visual Media, including eRENA programmer-researchers and CG animators, together with a team from the Medientheater (comprised of the theatre's technical director, sound, lighting, and projection engineers), prepared the workshop environment in terms of computer and motion capture resources. In parallel, the workshop participants received instructions allowing them to prepare a series of preliminary "ice-breaker" exercises to initiate the encounter at the International Institute of Puppetry; these exercises involved testing and evaluating body-object dynamics across a set of formal experiments. A list of workshop participants is provided as an appendix to this chapter, together with the IIM and ZKM workshop teams.

Since the workshop constituted a reference point for ongoing research on extended performance event management and toolkit prototyping, the following text discusses the IIM-ZKM activity from this angle. An overview of the working environment precedes descriptions of three performance experiments focussed on the notion of technologically generated "theatrical doubles". These accounts lead to more detailed technical analyses by Michael Hoch and Bernd Lintermann, who respectively investigated vision-based and magnetic capture-based systems

during the workshop. The chapter concludes with a summary of some key features for the design and management of events in electronic arenas, and of content creation for them, which emerge from our work.

On the basis of our experience with the workshop, design principles for a generic “black box” have also been identified. This technology supports event design and management and content creation by offering readily storable, transformabable, interchangeable semantic mappings for bodily-controlled computer graphics to be used in live performance in electronic arenas. The prototype system, currently under development by Bernd Lintermann at the ZKM, is described in Chapter 2 of this deliverable.

Figure 1.1: Experimentation with simple geometric objects
Left to right : Kirk Woolford, Gisèle Vienne, Ramon Rivero, Ariane Andereggen.
ã *Jacques Sirot*

1.1.1 Technology-free performance experimentation at the IIM: playing with constraints

A first technology-free week at the IIM was devoted to relatively systematic exploration of body-object relations, using simple geometric objects (spheres and cubes). Various devices which could be broadly termed prostheses or interfaces were employed to impart different kinds of motion to human actors and objects (elastic and a harness to suspend figures in space for aerial movement, roller skates and attachable wheels to obtain gliding movement, sticks and strings to mechanically push and pull figures, etc). As the performers explored a range of dynamic behaviours, they became attuned to subtle differences in movement qualities, and to the drama

that could be forged by sheer confrontation between "species" of figures characterised by their respective locomotor activities. Group actions were developed where performers animated and identified themselves with cubes and spheres, made mobile by use of wheels or elastic, or simply carried or dragged. The task of imparting legible human energies and dynamics to these geometric shapes provided a first approach to mapping issues, and raised the question of whether the actor/animator needed to be visible, for his/her vital presence to be felt and conveyed by the manipulated object. This remains a key issue in the extended performance arena, where physical and/or metaphorical distances between real and CG bodies (visual or auditory) are technically regulated and mediated in innumerable ways. Ultimately, the nature of the mappings between these two realms (physical/CG) translates artistic choices and coherence.

During the second technologically-equipped ZKM phase of the workshop, it was planned to use movement tracking techniques including the Polhemus Ultratrak motion capture system, where computer graphics figures are driven by the movements of up to twelve sensors attached to their physical animator(s). Since the sensors are affixed to cables connected to a processor, and hamper body movement, we simulated these constraints during the first workshop phase. At the Puppetry Institute Theatre, movement sequences were limited to a stage area corresponding to the working range of the Polhemus system, and moreover rehearsed with physically tethered performers. Plastic-coated clothes-line wire was cut into lengths corresponding to the sensor cable lengths, these ersatz cables being bunched and suspended from one of the theatre beams, to prefigure as closely as possible the set-up that would be encountered at the ZKM Medientheater. The performers discovered the traps of certain kinds of motion in this encumbered situation; gestures were adapted and reworked to avoid snares, and untrammeled actors assisted cabled performers by discreetly guiding and preventing risky movements.

Deliverable D7a.1 documents the difficulty performers encumbered with immersive equipment experienced in the inhabited television experiment *Out Of This World.* The performers had little opportunity to rehearse with the equipment in advance of the immediate run-up to the performances themselves. By experimenting with low-tech simulacra of hi-tech equipment, our workshop participants were able to anticipate the rigours of encumbered performance that much better. This suggests the importance of mock-ups and other simulations in the preparation of events for electronic arenas.

With practice, the performers were able to execute increasingly complex, intricate sequences without becoming entangled. All kinds of configurations were tested, ranging from an individual wearing or carrying the sensors attached to his/her own body, or to an object he/she manipulated, through to a group of performers bound together with elastic (or stockings), the "sensors" being positioned on various parts of this multiply-limbed, collective body. Since motion capture can be used to animate an infinite range of non-human computer graphics morphologies, much energy was devoted to building strange collective shapes and dynamics, in order to prepare for the creation of computer-generated "partners" free from the limits of conventional human gesture. Training to heighten a sense of collective rhythm—the essence of "sannintsukai" or three-man technique employed in traditional Japanese bunraku puppetry—was emphasised. By testing movement sequences in this manner, we were able to identify and develop proficiency within gestural registers which were then readily transposable to the motion capture situation. This did not mean that we wished to preclude movement that was impossible in the encumbered capture environment, since live performance does not necessarily have to exclusively resort to any single technique or technology. On the contrary, gesture that is refractory to motion capture was also

substantially worked on as an integral part of our exploration of technologically extended performance, our identification of what does and does not lend itself to technical extensions, and our research into how to create interesting performance by effectively hybridising raw and technically mediated action.

Figure 1.2: Tethered practice session - premises of the Chorecalligraphy *number*
left to right: Susan Kozel, Cyril Bourgois, Ariane Andereggen, Gisèle Vienne
ã *Jacques Sirot*

In addition to working with geometric objects, we explored the specific dynamics and theatrical implications of a wide range of puppets (string, rod, glove, bunraku, etc). This experimentation raised questions of how to adapt handling techniques, and triggered reflection on how sometimes modest changes in scale can dramatically transform the puppet-puppeteer relationship. Handlers practised technically accommodating and theatrically relating to the dynamics of figures of different sizes, made of different materials, endowed with differently articulated mechanisms. The intimate feedback loop that puppetry is based on, where gestural behaviours of figure and handler constantly draw on and transform one another, was often dramatically evident in the course of improvisation sessions with our extended troupe of puppet-actors. As the workshop participants became more familiar with each other's techniques and more engrossed in questions of the dynamics of human and inert bodies, zones of interplay between actors and puppets intensified. Several experiments tended towards marionettisation of human performers, and interchangeability of or gestural mimicry between actors and puppets was rife. This promiscuity

between the two realms subsequently proved to be a valuable acquisition in the ZKM environment, where another species of stage partners—namely computer graphics elements—placed heavy new demands on the performers' capacity for self-projection and transfer of gestural energies.

Figure 1.3: Manipulating puppets/people - premises of Monte Carlo & Poisson numbers
Left to right : Ariane Andereggen, Susan Kozel, Gisèle Vienne
ã *Jacques Sirot*

1.1.2. Technologically-extended performance experimentation at the ZKM: scrambling screen-stage boundaries

Our goal at the ZKM was to explore theatrically diverse situations involving different kinds of screen and stage action, in a relatively conventional situation (i.e. ultimately in front of a passive, seated audience). We first sought to reconfigure stage space to escape the dominance of the central projection screen in the Medientheater, and thus be able to test relationships between live performers and their computer graphics counterparts in a wide range of spatial situations. There were limits to this reconfiguration, since our main experimental technology was motion capture, which in our trial configuration was essentially screen-bound and, which as mentioned above, introduces its own specific spatial and technical constraints: respect for the sensor range, avoidance of metal interference with magnetic capture system, unencumbered camera view for the vision-based tracking system, etc. The magnetic capture source, calibrated prior to our arrival,

had to stay in a fixed position, as did the overhead tracking system. These considerations, together with the necessity to uphold good computer graphics visibility, restricted our use of the performance space (on these and related issues, compare the compromises made in the production of *Murmuring Fields*, Deliverable D6.2). Our goal was to explore theatrically diverse situations involving different kinds of screen and stage action, in a relatively conventional situation (i.e. in front of a passive, seated audience).

First, we suspended the Polhemus cables approximately 3 meters above the stage floor to keep the ground area maximally free for moving actors, thereby reintroducing the configuration we had already experimented during the previous technology-free week. We re-rigged our harness to allow performers to intervene in unusual aerial positions. Setting up this harness turned out to be a major step in reaffirming the primacy of acting bodies in a space where the frontal screen tends to dwarf if not eclipse live action. The crudely mechanical shadow thrown onto the screen by the harness rope and hook somehow anchored—and tamed—its surface.

Two fiber-glass scaffolding units were wheeled into the theatre to break up the flatness of the stage and, yet again, to allow performers to get off ground level; these units measured about three meters high, with practicable intermediate and upper levels. An additional useful attribute of these platforms was the fact that, contrary to metal scaffolding, fiberglass poses no problems of interference with a magnetic capture system (we had to avoid using the sensors too close to the rear wall of the theatre, with its heavy metal fire door). One of these units was used for stacking the Polhemus processing unit, and to house an improvised velcro “rack” for sorting and maintaining the sensors; the other was kept free for performance purposes.

The central projection screen, maximally deployed on our arrival, was raised to maintain a screen-free area about 2.5meters from the ground, such that performers visible from any point in the Medientheater were acting in a discrete space, without being inadvertently caught up in images being projected onto the large screen. At least initially, to investigate the different qualities of the performance areas created by the available technologies, and to sound their demarcations, we wished to respect separate stage zones for live and computer-generated action—even and especially when these were simultaneous. By upholding this separation, we were also attempting to draw lessons from the previous year’s opera production (cf., from Year 1 of eRENA, Deliverable D2.3, notably Section 2.6, “The 2D-3D Screen-Stage Gap”), where lack of clear codification governing the use of screen and stage space had sometimes led to uncomfortable, seemingly arbitrary interplay between human and computer graphics actors. Having literally diminished the surface of the main screen, we stood two small rear projection screens (approximately 2 meters by 1.5 meter surface, set on wooden bases raising them 10cm above ground level) on the stage floor, each with its own projector, slightly angled inwards and placed on either side of the large central screen. It was easy to switch between these, or to use all at once. Combinatory tactics attenuated the Medientheater’s conventional cinema overtones, introducing new flexibility into the stage setting and use of graphics. The small screens were on the same level as much of the action, and their more human scale made it easier to grasp and dramatically play with the relationship beween projected images and live actors. The floor was also used as a projection surface, to disintegrate or rather disaggregate the simplistic “screen versus stage” boundary.

Another ploy to disrupt the dominant frontality of a theatre apparently more suited to projections than to live performance was use of one of the technical control rooms to stage a brief episode in the course of the final presentation. In addition to the conventionally located

projection room, which looks onto the main screen from a window high up on the rear wall, the Mediентheater possesses two other control rooms, built into one of the lateral walls at the same height. All the technical facilities are accessed via external stairs, and their windows overlooking the house usually go unnoticed because of their height and the discreet lighting employed by the technicians working there. We did not need these spaces for technical purposes, but discovered that effectively oriented lighting in one of them allowed strong shadows to be projected through the window onto the theatre's main screen. Intensely lit, this space served as a kind of puppet booth for a sudden, Petrouchka-type apparition, as a small rod figure swiftly traversed the window, curiously observing the public below. This figure threw an immense black shadow onto the big screen, and there was instant dramatic tension between the physical puppet's whimsical deambulation across the glass screen, and the ominous movement of its gigantic shadow double across the screen in the theatre.

Other experiments focussed on the borderline zone in performance space where motion capture is not quite operative, i.e. the area just out of tracking range. This area is imbued with its own peculiar theatricality: while it may be a perfectly visible part of the physical stage, at the same time it exudes latent dramatic force similar to that manifest by the wings or footlights, because of its contiguity with the active capture region. We were striving to better differentiate various kinds of performance space, concentrating on the notion of boundaries and areas dramatically and specifically charged by certain technological tools. Wilfully placing and framing stage action – using lighting or props to "develop" a scene in the way a chemical developing agent reveals a photograph – is an essential task in theatre. Indeed, chase projectors and highly focussed spots, and the "theologeion", a physical platform reserved for the gods, built as a stage element in ancient Greek tragedy, figure among the numerous precursors of modern technologies used to demarcate and fire specific performance loci (cf. Deliverable D7b.1, notably chapter 3, the notion that an electronic arena may have different spatial regions each with a different potential for interaction is also explored in Deliverable D6.2, and from the first year of the project, in Deliverable D2.3).

Technological choices were hinged on theatrical concepts and projects that were deliberately kept very simple, hence the unabashed use of low-tech configurations alongside more sophisticated systems running on the Onyx. Our experiments were necessarily short-lived and, in final form, were still rather rough, but it was considered important in the workshop context to test a broad spectrum of situations and relationships, rather than to polish a more limited number of theatrical sketches. In addition to testing various spatial configurations, situating human actors differently with respect to electronically mediated co-actors projected onto screens, floor, or draped fabric, we explored the theatrical relationships that arise between live performers and different kinds of computer graphics. Non-figurative motion capture driven imagery was explored, together with more-or-less realistic, figurative graphics, as a function of the software employed. The interplay between actor and image, and their combined dramatic impact, changed radically with these changes in display.

1.2. Technologically Generated Theatrical Doubles

1.2.1. Multiple screens to multiply avatars: *Ubu Roi*

A sequence from *Ubu Roi* performed by Cyril Bourgois, a puppeteer specialised in fairground and street theatre, was developed in the Mediентheater to exploit the multiple screen

combination. At the Charleville Puppetry Institute Bourgois began practising this sequence by playing the dialogues wearing the Père Ubu glove puppet on the right hand, and replacing the left-hand Mère Ubu glove puppet with a palm camera. During the couple's tirades, black-and-white footage captured by the Mère Ubu camera/viewpoint was relayed by a video monitor. Père Ubu's vulgar mishandling of his equally unlikeable partner were thus seen through her eyes, as the camera effectively conveyed his boisterous movements. This simple situation seemed to withhold much potential in terms of interplay between physical and projected actors, so we decided to create a computer graphics model of Mère Ubu, animated with Polhemus sensors borne by the camera hand. The further layer of prosthetisation thus implied, motion capture sensors being combined with camera-conveyed vision, afforded rich development of a "virtual puppet" concept. Sabine Hirtes, computer modeling/ animation expert at ZKM, used Softimage to create a fairly simple polygonal model of Mère Ubu, appropriate for real-time performance. The cloth texture of the puppet's costume was scanned in, painted texture was used for the face, and a simple jointed skeleton was created, for connection to the Polhemus sensors via the Ultratrak driver.

Figure 1.4: Ubu Roi - *Cyril Bourgois interacting with Ubu camera image (left screen), Sabine Hirtes' Mère Ubu CG avatar animated by Polhemus sensors (centre and right screens), affixed to Père and Mère Ubu glove puppets*
ã *Jacques Sirot*

We used the Medientheater's multiple screens to play on the protagonists' multiple representations : the puppeteer was visible, left centre, holding the Père Ubu puppet and the prosthetised version of Mère Ubu, equipped with palm cam and Polhemus sensors; these devices

were stitched onto a white glove to form an oddly anthropomorphic shape. The left-hand rear projection screen showed black-and-white images of Père Ubu relayed by the Mère Ubu camera. The right-hand rear projection screen showed the coloured computer graphics model of Mère Ubu, animated by the Polhemus sensors. The latter model was simultaneously projected onto the large central screen as well, to weight the stage action and override too-predictable symmetry. A dramatically interesting counterpoint arose between the real human actor, the physical Père Ubu puppet and physical Mère Ubu puppet/prosthesis (camera and sensors), live black-and-white footage of the Père Ubu puppet, and the computer graphics Mère Ubu puppet animated in real time. The graphics figure on the large central screen towered over the human actor, but complexity of this representational system dispelled any sense of simplistic rivalry between the two. Indeed, it was quite clear that the gigantic virtual puppet was not only controlled by the puny human actor, but was moreover tyrannised by the little Père Ubu puppet, whose lunging, taunting behaviour was vividly conveyed by the black and white film. Jarry's truculent dialogue lent itself admirably to this power struggle between physical and projected actors.

1.2.2. Single screen to multiply avatars: *Shadow Puppets*

Another experiment exploiting the theatricality inherent to coexisting registers of presence involved a mix of simple shadow puppetry techniques and real-time black and white projection. For this piece, called *Shadow Puppets*, Gisele Vienne and Cyril Bourgois used two string puppets, the palm camera being affixed to the head of one of these. The puppets were manipulated behind one of the rear projection screens, such that the shadows of their physical bodies—and those of their puppeteers' hands and feet—coexisted on the screen with filmed images relayed by the puppet with a camera head. The rear projector light beam conveying the filmed images was judiciously employed by the puppeteers, to ensure an indecipherable mix of real shadow and cinematographic projection.

Striking infinitely receding effects occurred when the camera-headed figure filmed the real screen shadow of its puppet counterpart, momentarily filling the screen with dozens of identical shadow figures. Sudden intrusions into the screen space of the puppeteers' bare hands and feet, decidedly human although viewed as monochrome shadows or projections, further disrupted our sense of the boundaries between physical actors and immaterial projected actors. The puppets were manipulated on a low wooden platform to bring them up to the screen base level, and their movements resounded eerily, setting the theatrical action in an acoustically, as well as visually specific locus. The entire sequence had a black-and-white early thriller atmosphere, with its zooming close-ups and grainy anamorphic images, its layers of ambiguous shadows—some corresponding to physical substance, others technically mediated—and the resonance of wooden puppet feet on a wooden stage floor. Although the configuration was extremely simple (no computers were used), this dramatic mix of hybrid images and shadows pointed towards higher end but conceptually similar experiments, integrating live video footage into computer graphics displays, and/or relaying live camera images from and back into the Web for example. Hybridising physical interaction and CG material is of course a prominent theme throughout eRENA (cf. especially Deliverables D7b.1, D6.2, D6.3 and the account of the Round Table production support environment in this deliverable).

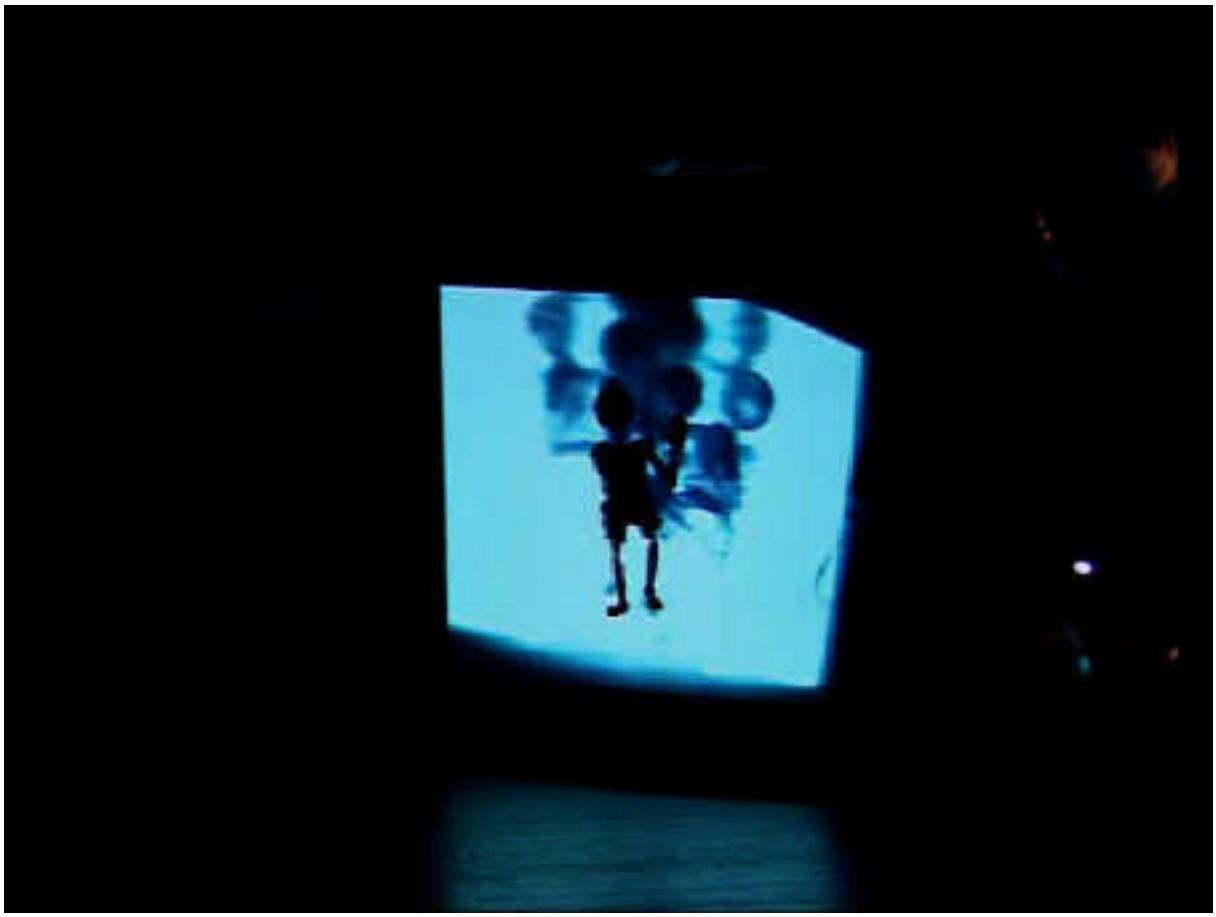

Figure 1.5: Shadow Puppets - infinitely receding effects obtained by integrating a camera head into one of the puppet figures
Manipulation by Gisèle Vienne, Cyril Bourgois; rear projector handled by Kirk Woolford
ã Jacques Sirot

1.2.3. Low-tech pointers to theatrically effective high-tech: *Monte Carlo*

Explored interactions between live and projected actors ranged from shadow theatre, light-triggered graphics interactions with performers, mixes of live actors, real-time film and real-time motion capture driven graphics, through to interplay between physical and motion capture driven figures, or between physical and chromakeyed characters in a standard playback situation. Hence, for example, a stark relationship between the human and projected performer was procured in the simple playback duo entitled *Monte Carlo*. In this piece, the pre-recorded act of an aging puppet songstress, animated by actress Ariane Andereggen and filmed by Sabine Hirtes in the ZKM blue studio, loomed over the actress perched precariously upstage on a bar stool. The red feathers floating in the black film background, vestiges of a feather boa that had seen finer times, added pathos to Ariane Andereggen's presence as she toyed with the same red feathers while delivering her depressing *Monte Carlo* rendition. Of interest here was the discrepancy between the screen and stage characters : the deliberately thwarted moments in playback were dramatically powerful, as the filmed diva thereby asserted her autonomy with respect to her human counterpart, and vice versa. This same observation recurred constantly in the course of workshop experimentation, particularly in the context of discussion on technologically high-end mapping issues. Research with the vision-based system (cf. below Section 1.3) and with the xfrog-based system (cf. below Section 1.4) led to similar conclusions from the theatrical standpoint: a rigorously systematic

relationship between a screen and human figure generally lacks dramatic interest, whereas differential zones, areas of “slippage” between the two species of players, withhold inherent pathos. In *Monte Carlo*, as in *Shadow Puppets*, a low-tech approach provided valuable theatrical pointers for future extended performance research.

Figure 1.6: Monte Carlo - *Ariane Andereggen performing playback*
ã *Jacques Sirot*

1.3. Specific Experimentation with the mTrack Vision-based Interface

For tracking performers' movements in an unencumbered way, Michael Hoch's vision-based interface system was used. This system is based on colour segmentation, motion detection and blob analysis. mTrack is divided up into the recognition part consisting of the image processing system, a server program, and the library front-end with the application program (see Figure 1.7, Hoch, 1997, 1998).

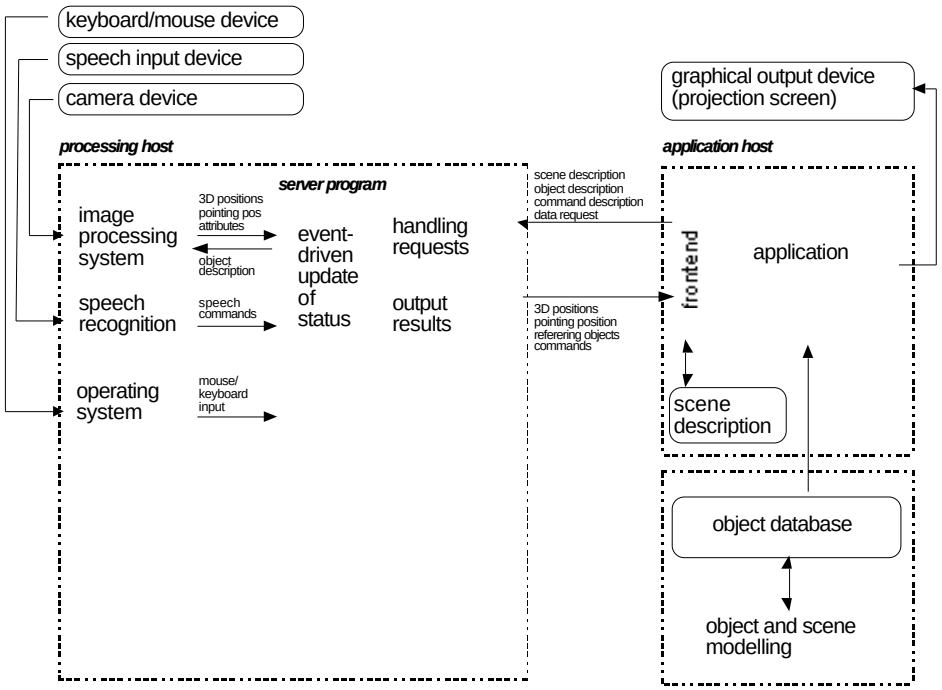

Figure 1.7: Architecture of the mTrack tracking system

The image processing system tracks the user via one or two video cameras. During initialisation it receives a description of the objects to be tracked. Thereafter, it continuously sends position data or other calculated information pertaining to the segmented objects, e.g. the performers' current position, to the server program. The server program connects an application with the image processing system. It updates the current states by an event driven loop. Upon request it will send data to the application continuously. The library front-end is a collection of methods that supply basic server communication. It reads a description of the objects to be tracked from a text file on the application site. The application program implements the methods to perform the desired task. It uses the scene description and the library front-end to communicate with the server program. The application may run on a different host than the server program. In our case the application defines the real time device that feeds data to the Maya motion capture architecture, which then responds by generating the graphics and animations.

1.3.1. Tracking coloured regions

To obtain update rates higher than 10 frames per second, essential for establishing direct feedback between performer and graphics (Stary, 1996), we have to choose a compromise between sophisticated tracking algorithms and simplicity to achieve the desired frame rate on standard PCs. A simple and robust approach consists of tracking coloured regions. If objects to be tracked are painted with colours that are distinct from each other and are not present in the background, a simple segmentation algorithm based on thresholds in the UV-colour space together with a blob analysis is sufficient to reliably track these objects at a high frame rate.

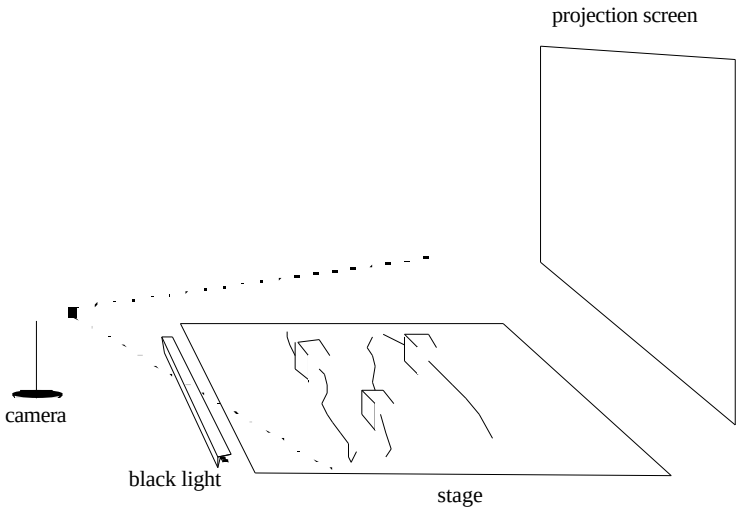

Figure 1.8: Setup for tracking fluorescent cubes

In order to test the tracking of coloured regions in the extended performance workshop, we took the wooden skeleton cubes built during the initial Charleville phase of the workshop, and mounted white lycra fabric onto these cube frames. The cubes where then sprayed with fluorescent colors (yellow, green, and red). Thereafter, the cubes were connected to long elastic bands so that the performers could animate and „play“ with these objects, generating aerial choreographies with them (an activity which had been extensively worked on at the International Institute of Puppetry). A camera connected to the mTrack system observed the scene and recognised the xy-positions of the different cubes. The position data was then used to drive a Maya particle animation in real time, shown on the large projection screen behind the performers. This experiment gave rise to a simple, decorative form of kinetic theatre, where the human actors were largely eclipsed by interplay between the fluorescent physical cubes and their electronically mediated particle “reflections”.

1.3.2. Motion detection

The piece called *Contours* simultaneously employed projections of luminous forms on the floor and on the central screen, such that traces of two dancers “swimming” across a predefined rectangle of trackable stage space could be seen on the ground, appearing as bright white silhouettes etched round the crawling bodies, and were also visible on the screen hovering over and behind this rectangle. This configuration was deranging, since the figures appeared to be both

pinned down and floating, grounded yet free to roam into a seemingly inaccessible performance area, to move by technological stealth into another dimension.

Another approach was adopted to seamlessly integrate virtual and real space by using motion tracking on the stage and simultaneously projecting the resultant graphics onto that same stage space, occupied by the human actors. Here we had to carefully adjust the lighting conditions so that the projected imagery would not interfere with the tracking algorithms. A series of luminous apparitions dubbed *White Series* was created with this technique, though here was limited to floor projections only; we used black lighting and white objects (gloves, book, umbrella, white stockings, white puppet) integrated into the choreography of the performance. The result was a technologically upgraded kind of "black theatre", which included such standard cabaret tricks as bodiless hands (i.e. the white-gloved hands of a performer dressed in black), but which drew new theatrical magic from the ghostly white shadows formed by the CG particles. The lighting level of the projected particle animations was adjusted to uphold a high contrast between the white objects under black light and the projected images.

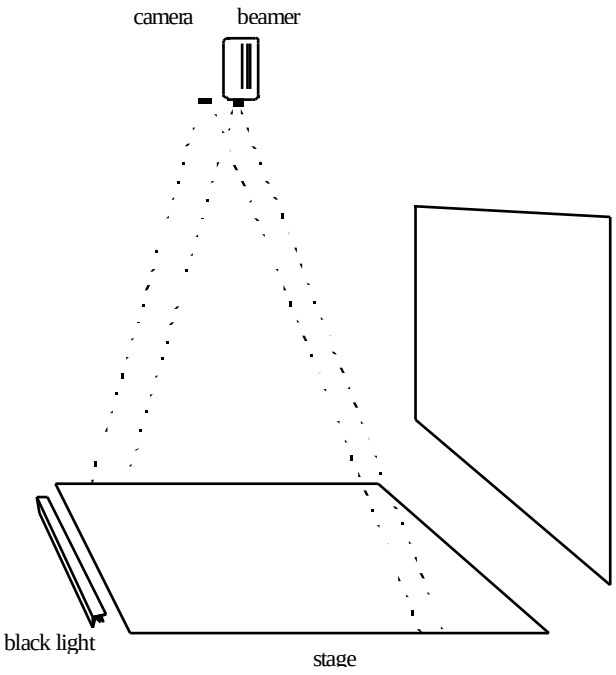

Figure 1.9: Setup for motion detection (used in White Series*)*

For motion detection we used the algorithms outlined below:

```
PROCEDURE motionTrack

WHILE(true)
  // acquire current image
  current = get_next_image()

  // remove Gaussian noise
  low_pass_filter ( current )

  // perform background subtraction and binarize
  image = background - current
  binarize ( image, less_or_equal, 15 )

  // extract the relevant moving parts
  Do ( 3 times )
   dilate ( image )
  Do ( 2 times )
   erode ( image )
  thickening ( image )

  // analyze result image
  // exclude small areas and scattered regions
  blob_analysis ( image )
  exclude blobs with area <= 550
  exclude blobs with compactness >= 5
  extract_position_data()

  // send data to server and update background image
  send_data_to_server()
  background = 0.9*background + 0.1*current
  ENDWHILE
END
```

First, each image is filtered by a low-pass spatial filter that reduces Gaussian random noise (and high-frequency systematic noise). The filter replaces each pixel with a weighted sum of each pixel's neighbourhood. For detecting motion, we use a background subtraction procedure that updates the background with 10% of the current image, i.e. the background is continuously updated and a moving object will become background after a while (approximately 2 seconds). After background subtraction, the resulting image is dilated three times and eroded twice. Binary dilation of an object increases its geometrical area by setting the background pixels adjacent to an object's contour to the object's pixel value. Erosion will do the opposite, i.e. set the contour pixel of an object to the background value. A combined dilation and erosion operation will close small holes in the object. Here, we not only want to close holes but also try to merge adjacent regions to obtain a single connection region for the moving object. Therefore, we use a 5x5 kernel for erosion and dilation, followed by a thickening operation that further increases the geometrical size of the object. Next, a blob analysis procedure is performed and smaller areas are excluded from the calculation as well as areas with a compactness of greater than or equal to 5. The compactness c of the segmented region is defined by:

$$c = \frac{p^2}{4\pi A}$$

whereby A denotes the area of the regions (one Pixel corresponds to an area of 1), p denotes the perimeter, that is calculated from the pixel edges including the edges of holes. For a circular region without any holes, this formula leads to a compactness of $c = 1$. By excluding areas with a value of greater than or equal to 5 we focus on round shaped objects, i.e. areas that are likely to be detected when a person is observed from above. Finally, we extract the position data, i.e. center information about the detected blobs, and send the data to the server.

The described algorithms can reliably track up to four people with a frame rate of > 12 fps. Because of the particular background algorithm, a moving object will not be tracked if it stands still for more than approximately two seconds. This also means that the resulting graphics that are projected on the stage floor disappear after that time. For the workshop scenario, this characteristic was integrated into the choreography of the piece that used this technology: the "black magic" atmosphere of the *White Series* was enhanced by this gradual vanishing of white shadows.

1.4. Real-time Animation and Motion Blending

Real-time animation of computer generated imagery often needs extensive programming and development time. The problem not only resides in creating an appropriate geometry (2D, 3D or even abstract), but also in managing data structures and supplying the appropriate driver modules for real time interaction. With Maya software by Alias Wavefront, a 3D modeling and animation tool is available that allows a great amount of real time interaction and visualisation (Alias Wavefront, 1998). Furthermore, a script language (MEL) can be used to programme complex relationships between an external device driver and geometric attributes or variables of the animation. Another feature of the Maya programme allows the user interface to be switched off completely then remade visible again by a simple keystroke. By hiding the user interface, one obtains a single window on the screen, which gives the impression of a custom graphics application. Switching between those two settings proved to be very useful for a development and testing phase.

For the workshop, Maya was used to create a particle animation system that responded to different device drivers (vision-based tracking data and Polhemus magnetic sensors). Magnetic sensors drove the computer graphics for the piece called *Poisson*, where Gisèle Vienne manipulated a human-amphibian string puppet from the upper stage of the fiber-glass scaffolding platform, engaging the little figure (about 20 cm) in an ethereal floating duo with dancer Susan Kozel, suspended from the harness. A long piece of white fabric was draped from the platform and from three tripods momentarily placed on the stage, used as a projection surface for abstract particle graphics controlled by the Polhemus sensors worn by both puppet and dancer. The makeshift screen in its draped organicity—the term "organic screen" was aptly coined by Michael Hoch to describe this configuration—gave unfathomable depth to the computer graphics, setting up a very different kind of relationship between these elements and the human performers and puppet. The projection of images conventionally bound to two-dimensional display systems—cinema screens or monitors—onto surfaces of ostensibly three-dimensional objects is a curiously strong source of drama, which is no doubt inherent to this unholy wedding of unlike species, this strange union of 2D and 3D elements (Tony Oursler's installation figures, solid white dummies onto which filmed, animated faces are projected, precisely target this disturbing dimensional encounter, see Deliverable D4.1 from eRENA Year 1).

A second animation devised during the workshop, for an experiment called *Time Layers*, created a simple motion blending technique that could be tested by a performer in real-time. The following diagram describes the components of the setup:

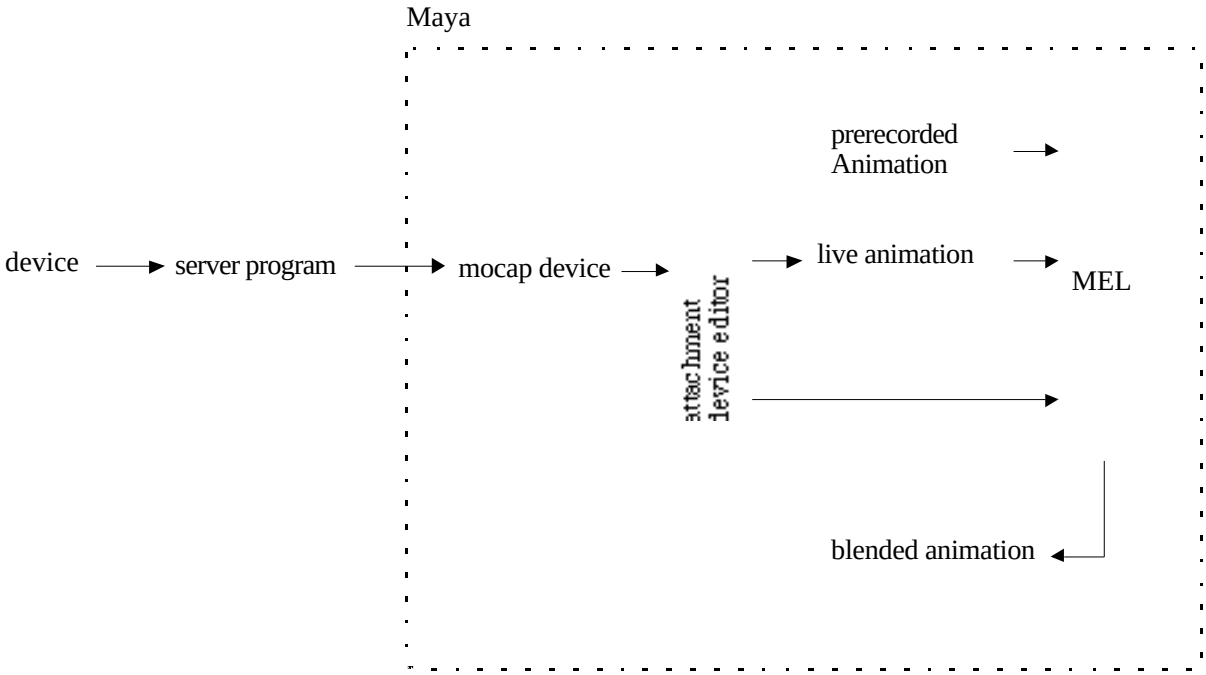

Figure 1.10: Maya setup

The input data for the Maya setup was either the Polhemus sensor system or the vision based motion tracking system mTrack described above. The connection between the actual device data and the Maya system was established by a server programme for each device (this comprises mainly a C program that implements some functions for connecting to Maya and registering certain variables, and for ensuring the continuous updating of these variables by the device data). In Maya, the server programme needs to be registered as a mocap device. Thereafter, the user can define the links by using the attachment device editor. This allows the attachment to be established between a device axis and an objects attribute, e.g. coordinates, rotation values, deformation values, or generic variables. This setup has been used, for example, to create the prerecorded animation and is now used to drive a live animation of an abstract figure. Using a set of simple MEL scripts, the two animations are blended and thus yield a third (blended) animation. All three animations are visible to the performer, who can then react in real-time to the algorithms and create an online choreography. The nature of the blending depends on the performer's movements. If he/she moves a lot, this increases his/her control over the blended figure; if he/she moves less, the pre-recorded animation takes „control" over the figure. Different settings make it either simple or hard for the performer to gain control.

We added a time-sensor to induce periodic switches between the two settings of „easy control" and „hard control" The interaction increased tremendously and was dramatically greatly enriched after this addition. The performer and computer scientists agreed that this interaction constituted an extremely interesting direction for future research. Although the workshop was too brief to

allow further investigation in this area, a technique exploiting settings which modulate the nature of the interaction between a human performer and computer graphics figure is clearly the source of powerful artistic potential. (This bears comparison with work reported in Deliverable D3.1 in Year 1 of eRENA. In the *Lightwork* performance described there, varying extents of 'algorithmic mediation' between performer activity and its effects on real-time CG and sound manipulation were experimented with. In the current deliverable, we later describe how different forms of control might exist for the manipulation of virtual cameras in electronic arenas.)

1.5. Specific Experimentation with xfrog and a Magnetic Interface

A series of experiments based on Bernd Lintermann's xfrog software, using the Polhemus Ultratrak magnetic capture system, showed the interest of developing readily mappable sets of parameters to graphically "interpret" real movement in a host of different ways, ranging from isomorphic or literal response (where gestural amplitude and velocity engender proportional changes in graphics), through to non-symmetrical or metaphorical response (where graphics are animated in non-isomorphic ways by movement input). It seemed useful to work towards the elaboration of generic mapping systems, and to refine controls allowing rapid switching between such systems. Two mapping mechanisms for Polhemus tracker data were integrated into the xfrog software before the workshop: 1) *Direct mapping:* tracker translation and orientation data were directly mapped onto xfrog components, and thus used to transform complex geometries as a whole. 2) *Indirect mapping:* the euclidian distance along one axis/plane, or in the space of a tracker, the distance to a defined origin in that space (the emitter), could be applied to arbitrary xfrog parameters, or even to additional dynamic constraints like speed of movement, using a spline for mapping. Software for controlling basic stick figure morphologies, and for extracting control data, was likewise developed before and specifically for the workshop. This software is standalone and communicates with xfrog via tcp/ip.

1.5.1. Metaphorical versus direct mapping: testing performer ease

In a piece dubbed *Cyberflora mutatis*, puppeteer Ramon Rivero was equipped with all twelve Polhemus sensors, to control the on-screen evolution of a huge computer graphics flower (central screen). The actual model used was created by Lintermann's *Morphogenesis* system, an artificial life work designed for "cultivating" constantly evolving graphics in a networked or installation context, rather than for use in a live performance setting (cf. Lintermann, 1997). *Cyberflora mutatis* was conducted as an experiment to determine how comfortably the human body can control a non an(ti/thro)promorphic figure with no relationship to physical human architecture. The pre-existing mapping of control values, ensured by sliders in the original *Morphogenesis* context, had to be replaced in the theatre context by control values extracted from Ramon Rivero's posture as the motion-tracked performer of the piece. The tracked sensor data was thus employed to control a human stick figure, from which high level control data, e.g. bending of the arms/knees, extension of the limbs, were extracted and used to control geometric parameters.

Figure 1.11: Cyberflora mutatis - *Ramon Rivero interacting with Lintermann's xfrog flower*
ã *Jacques Sirot*

The *Cyberflora mutatis* experiment focussed less on software per se than on attempting to identify the mapping mechanisms with which the puppeteer/animator felt most comfortable as a performer, particularly with respect to time attributes. A delayed mapping function, where control values provide a goal toward which the actual model parameters move, seemed to be the most appropriate conceptual solution for upholding the organic feel of the model: very slightly delayed response (approximately 0.25 sec) smoothed the movement quality of the growing botanic form. For Rivero, this temporal factor moreover translated almost physically during performance, acting as a kind of resistance to the puppeteer's movements akin to the resistance opposed by a solid material figure. But for this same reason, the lag had to be kept to a minimum: when it was too long, the puppeteer felt decoupled from the computer figure, and could no longer physically invest and inhabit the graphics "marionette" with the same conviction and the same sense of being in control. Experimentation showed that Rivero, who is proficient in numerous techniques from traditional puppetry as well as being highly skilled in 3D computer animation, was familiar with and simultaneously sought availability of two distinct kinds of mapping that one might describe as metaphorical and direct mapping. On the one hand he wanted to be able to use his own arm and knee bends to bend the organic "limbs", in keeping with a more metaphorical approach, and on the other hand he wanted complete, direct control to elicit a more "systemic" response from the graphics, and this meant minimising the mapping delay.

Another configuration which implicated a live performer and a “growing” graphic element from xfrog, together with image material culled from real bodies, was proposed by a piece called *Butoh Tree*. Poised stage left, with her bare back to the audience, dancer Susan Kozel gradually drew herself to upright position with a very slow sequence of butoh movements which, via her Polhemus sensors, slowly brought to life and “grew” a computer graphics tree displayed on the central screen. Then images of human hands were filmed and integrated in real time to the tree structure as gently waving “leaves”. The hands were those of Ramon Rivero, who remained kneeling stage right, facing the audience, throughout the piece, and only gradually introduced his hands into the camera field. The leaves then progressively disappeared as Rivero withdrew his hands, and the tree became deathly still again (Kozel, 1998).

Figure 1.12: Butoh Tree - *Susan Kozel animating the xfrog tree with Polhemus sensors; live video feed of Ramon Rivero’s hands employed to “leaf” the tree*
ã *Jacques Sirot*

As with Rivero for *Cyberflora mutatis*, Susan Kozel felt least inhibited as a performer when the bending of her arms caused a bending of the tree, and the bending of her spine caused a bending of the trunk. Since in this piece the tree was a metaphor for the human figure, and can be considered as being endowed with a similar architecture, the appropriate technique here consisted of using trackers as control points for a 3D-spline defining curvature of the trunk/branches. Mapping of the trackers to control points of splines was not implemented beforehand. Mixing of

two mapping techniques (the tree growth was defined using the distance technique, controlling a 1dof parameter as mentioned above) additionally required definition of the reference point (origin) for each individual tracker. Moreover, a separate scaling of the mapping was required on the three axes in space, in order to exaggerate up and down movement.

Figure 1.13: Stop the Train - Group capture piece using Polhemus sensors, with live video feed of moving fabric superimposed on overhead CG display
ã *Jacques Sirot*

1.5.2. Streaming live data into live performance

In *Butoh Tree*, the relationship between the real, filmed and computer-generated elements was totally visible and unambiguous, thus quite different to the *Shadow Puppets* mix. Indeed, the strength of the tree piece largely resided in the explicit links between the dancer, the puppeteer, and the tree image, while these three actors functioned in very discrete spaces. Sound was also used in our attempts to integrate recognisably raw, analogue material into this piece: an HF microphone was attached to the aluminum fabric that draped the lower half of Susan Kozel's body, to amplify the harsh, metallic noise of this material. Minutely controlled displacements of the dancer's strongly lit back generated this hyper-real sound, which took on metaphorical qualities, as though it were rendering macro-movements of muscle fibre. In terms of visuals, use of live video streaming in this piece endowed the tree with a poignant quality, bridging the gap between schematic hard-edged graphics and patently living flesh of the human performers.

Filming the hands was metaphorically and theatrically appropriate in *Butoh Tree*, with its austere, minimalist aesthetics.

Nevertheless, live video streaming need not call on such overtly human material to convey dramatic impact: one of our pieces, called *Stop the Train*, put earlier group experiments to good use as five performers lying on the ground, their hands and feet bound together with elastic, activated a large, semi-translucent veil thrown over them. The veil was studded with the twelve Polhemus sensors, which in turn animated a computer mesh displayed on the central screen above them (this is just a modern version of an old 19th century melodrama trick, where stage hands animate billowing cloths in order to sink ships, drown heros and villains, etc). The undulating physical veil was filmed during the performance, these images being grafted onto the graphics mesh which thereby acquired an uncanny materiality. Phagocyted real images endowed the otherwise cold wireframe graphics with strong theatrical presence and vivacity. The peculiar emotional tonality obtained when “real” images are injected into a CG field has already frequently been mentioned in the eRENA research context (cf. Deliverable D7b.1).

1.5.3. Control data extraction, control value abstraction

For *Stop the Train*, the computer graphics fabric was deformed using a bezier hyper patch consisting of 125 control points, groups of five control points being attached to a single tracker on the fabric. The definition of the hypercube requires normalized/offset tracker data and is sensitive to motion scaling. Each tracker movement triggered ripples on the surface of the fabric, which decayed over time. To achieve this, an abstract mechanism called "energy" was implemented, allowing energy to be pumped using a motion which decays according to a given curve. This energy influenced the amplitude of oscillating deforming balls. The oscillation was created by an abstract device inside xfrog which maps a counter over a sine function. Here again, mechanisms bearing semantic values were implemented in the animation software to process control values.

The duo called *Chorecalligraphy* created a grotesque contrast when pristine calligraphic shapes on the large screen were animated by a Polhemus-tracked duel between dancers Susan Kozel and Ariane Andereggen, who were literally bound to one another with stockings attached to hands, feet, and heads. In opposition to *Butoh Tree*, where the dancer rigorously kept to the same floor area, and where the reference point for the control points of the spline was consequently fixed (i.e. the anchor points of the whole tree), both actors in this piece were constantly moving around the stage space throughout the performance. To maintain integrity of the geometry, control points had to be moved relative to a reference control point (i.e. the sensor attached to the head). Consequently, movements traced by this reference control point triggered transformations of the overall geometry. Moreover, to enhance vivacity of the graphics, one of the limbs was given additional autonomous motion. This programmed asymmetry effectively skewed and dynamised the visuals, breaking the monotony of a too visibly “one-to-one” rapport between the graphics and the human actor.

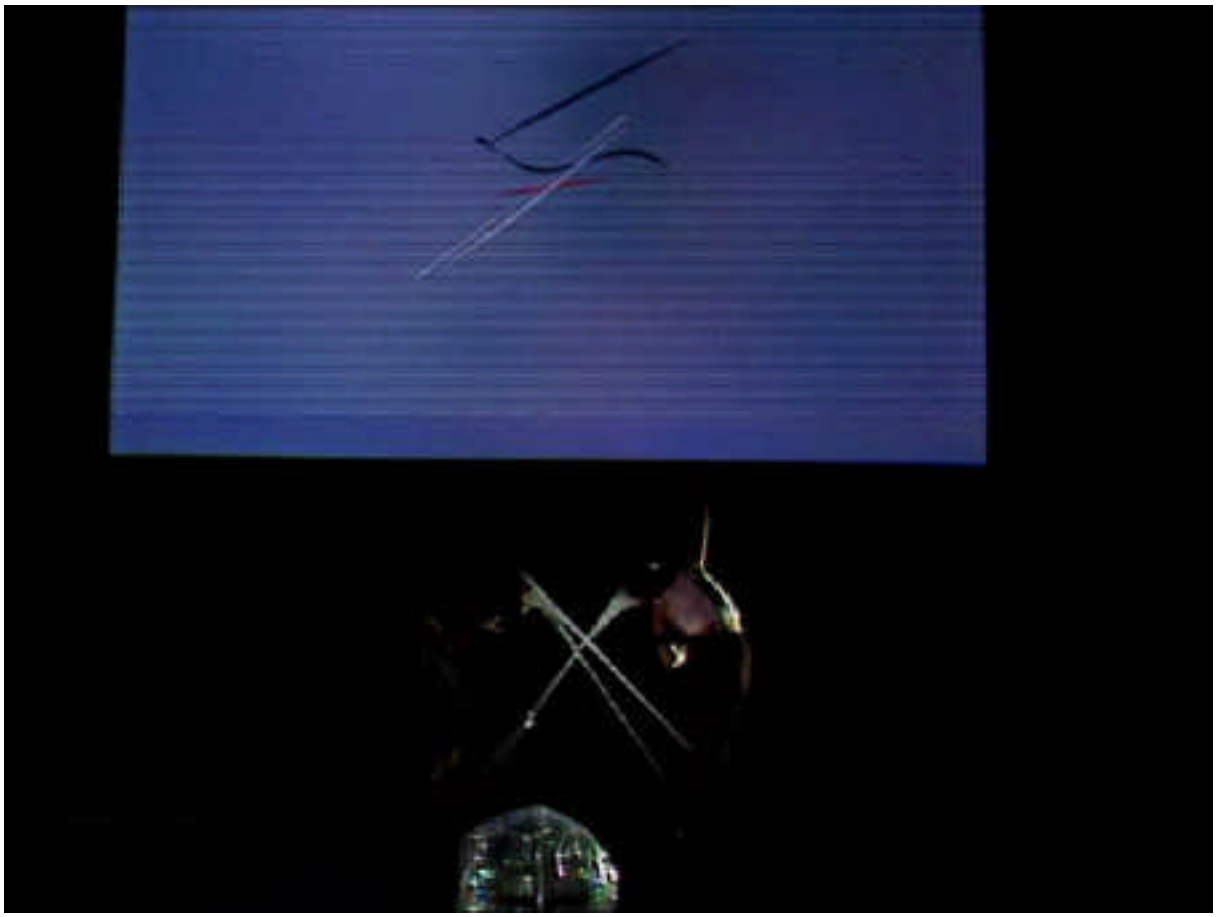

Figure 1.14: Chorecalligraphy - Ariane Andereggen and Susan Kozel wearing Polhemus sensors drive xfrog calligraphy display (magnetic capture source visible centre-front)
ã *Jacques Sirot*

An elementary motion capture configuration entitled *Clarinet Capture* consisted of a brief choreographic sequence executed by tracked performer Cyril Bourgois while playing a clarinet, his Polhemus sensors animating and transforming sets of simple screen geometries generated by xfrog. These fifties-style abstract deco graphics danced and mushroomed joyfully in response to the musician's whims. While such encounters between an actor and the images he/she drives on screen constitute a conventional motion capture situation, what proved particularly interesting here was the way the musical instrument generated and mediated a specific, poetic relationship to the visuals. The often irksomely narcissistic rapport between an actor and his/her motion capture driven computer graphics "mirror", i.e. situations where the actor is patently obsessed by his/her image—or just as obsessively pretends to ignore it—was avoided, since Bourgois' relationship to the display was mediated by an instrument which was itself artistically meaningful and expressive (cf. the problematic relationships between on-stage performer movement and on-screen avatar movement in *Out Of This World* as documented in Deliverable D7a.1, Chapter 6). In terms of the actual image processing involved, the same spline technique was used for *Clarinet Capture* as for the *Chorecalligraphy Tandem*, but in this piece, the work was designed so that the single computer graphics figure visible at the beginning spread out and multiplied, forming a circle during performance. Since the tracker yields only uninterpreted data, a keystroke on the keyboard was used to change the system state.

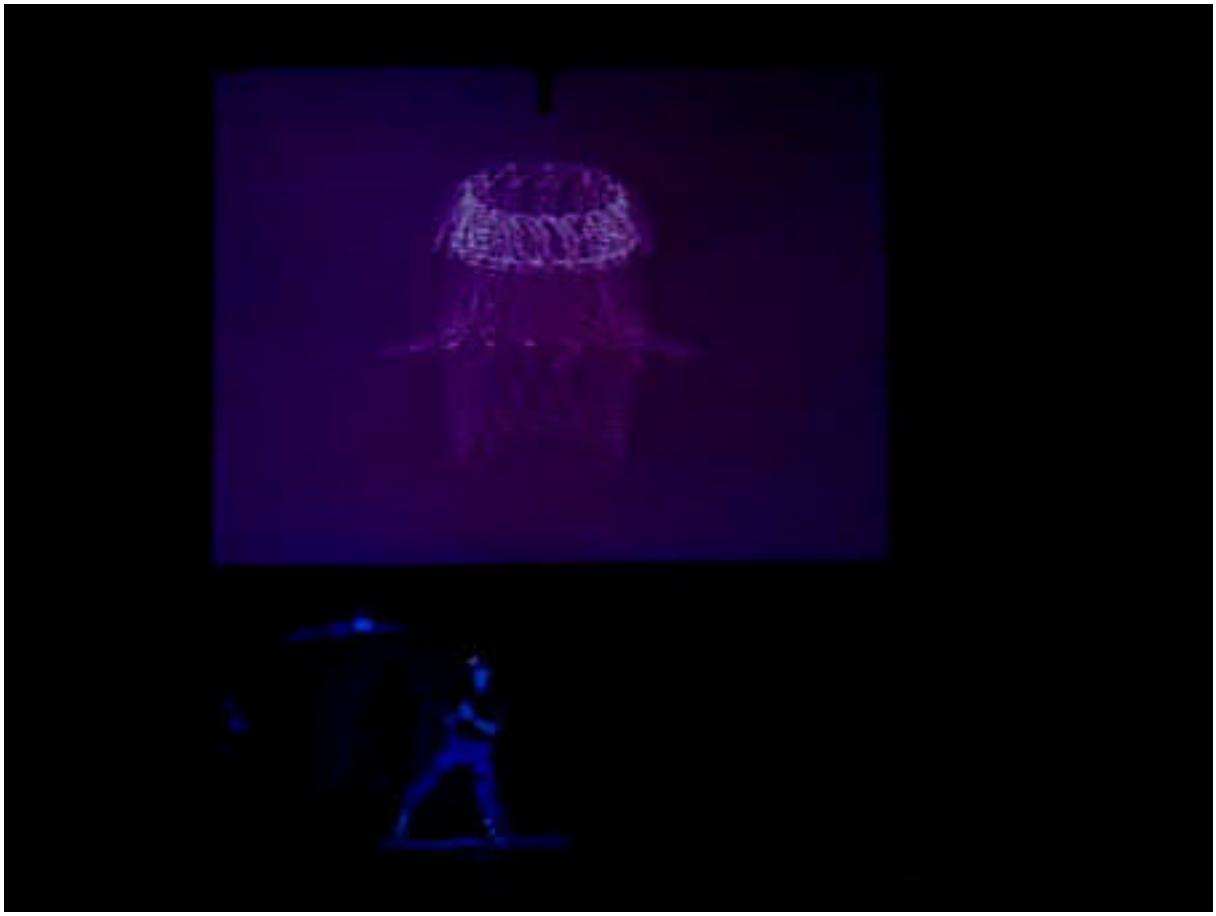

Figure 1.15: Clarinet capture - *Cyril Bourgois driving xfrog generated graphics with Polhemus sensors* ã *Jacques Sirot*

1.5.4. Making data dance: (semi-) autonomous mapping switches

In many contexts, it would be an ideal for the performer to be able to command the software in order to bring about a transition between two graphic situations. In a more sophisticated choreography, the tracker semantics (technically speaking, the mappings), would change, and should likewise be controllable by the performer. The ZKM workshop performance evening consisted of little pieces, and consequently there was time to reload scenes and reload different mappings by hand during the breaks. The system employed here would not be possible in the context of a continuous performance, even if the mapping method were as abstract as the delayed mapping technique used for certain pieces described above. Indeed, a continuous performance would require smooth mapping transitions triggered by the performer, thus interpreted control data. A vital component of event design and management in an electronic arena, and in the creation of content for an event, is managing the mapping and re-mapping of input control data to values for parameters required by whatever interactive algorithms (CG animation, sound or whatever) form or influence the content of the event. In short, to present a coherent event in an electronic arena (rather than the series of short pieces punctuated by 'technical delays' which comprised the workshop performance), there is a strong requirement for support in managing

transitions and enabling the mappings which underlie a participant's interaction with an event to be systematically configured.

Since problems such as attribution of reference points, scaling of tracker motion, determination of dofs, dof reduction, scale adjustment, range selection using a spline, occurred in all of the pieces in which xfrog was involved, it would clearly make more sense to deal with these problems generically, by devising a single piece of software which delivers interpreted, semantic control values, such as those which were discovered in the stick figure software mentioned in *Cyberflora mutatis*. There seems to be little sense in patching the animation system, which is meant to deal with graphics, with additional features to imbue raw data with meaning. Mechanisms bearing semantic values, implemented in animation software to create, interpret, and map control values, could in fact constitute part of a separate software module, since ultimately graphics software simply deals with the end result of the additional calculations. As animation software is endowed with its own specific architecture, such mechanisms cannot be easily applied to any input device, but must be implemented via a device made operational at the software code level. This approach is currently undergoing development in Lintermann's "Black Box" toolkit (see Chapter 2), which draws extensively on the real-time event management lessons gleaned during the 1998 eRENA workshop: *Real Gestures, Virtual Environments.*

1.6 Conclusions

In Deliverable D2.3 in Year 1 of eRENA, we reported on a number of the difficulties which had been encountered in enabling technologically extended artistic performances to take place. In particular, we analysed our experience of a "virtual reality opera" staged at the ZKM in 1997 in which real-time CG (projected to a large rear-screen) and performer-tracking played a role. A number of issues were identified concerning the problematic relationship between stage-space and large projections and in terms of making real-time interactivity legible for the audience while offering the performer opportunities for creativity. The August 1998 workshop directly served as a platform for investigating these issues further and practically working through possible solutions. One crucial issue is how to define mapping principles governing the interaction between computer-generated percepts and human actors. To what extent does theatrical effectiveness depend on establishing a recognisably analogous or isomorphic relationship between the flesh-and-blood figure and his/her electronic "shadow"? Conversely, what new dramatic forms are likely to emerge if the latter electronic shadows are emancipated from their human sources, imposing themselves as autonomous, full-fledged stage partners? Our work with various species of actors and interactors—human, puppet, shadow, projected graphics—attempted to explore these questions, to ascertain actor response and ease in a wide range of theatrical situations. The final public presentation of the workshop activities moreover allowed us to test the theatrical coherence of these experiments, and legibility of the given dramatic registers, on an audience largely comprised of novices in the area of real-time computer technologies. The audience's enthusiastic response indicated that basic gestural and theatrical skills coupled with interactive technologies indeed form a promising new artistic arena.

To be more specific, we feel that a number of simple technical and theatrical strategems facilitated our work and these are worth highlighting here as we believe them to be of general utility in designing and managing events for electronic arenas and in creating content for such events.

(1) Initially exploring theatrical ideas either without technological support or with low-tech mock-ups proved to be enourmously valuable. In other contexts the value of low-fidelity prototypes for computer applications is well-known. 'Cardboard computing' is often advocated over application development in a computational environment, especially when novel experimentation is the order of the day and it is not clear what the constraints on user-requirements might turn out to be. Prematurely engaging in software development can be costly (especially if mistakes have to be undone later) and overly constrain creativity. On the basis of our experience with the workshop, when participants are planning the design of an event for an electronic arena and exploring content for it, there can be great value in low-tech simulations, including simulations of the constraints performers might experience in interaction with high-tech equipment.

(2) To create a fledgling electronic arena for performance, rather than merely adding in technology to faniliar settings, we adopted a number of strategies for disrupting conventional relations between stage, screen and performer. We experimented with multiple screens with varying relationships between them. It proved particularly engaging to vary the sense of scale conveyed in a projected image and vary the relationship between the size of a projection surface and the human-scale inhabited by the performers. In a multiple projection setting, to have at least some sources of image and domains of interactivity which were of a human-scale seemed of most creative potential.

(3) It is important to support variable relationships between human performers and any on-screen counterpart, whether that is an avatar or some CG entity under interactive performer control. A close coupling or direct mapping between performer gesture and technically generated outcome is sometimes required but, on other occasions, a more metaphorical relationship is more satisfactory. Sometimes the relationship between, say, performer movement and on-screen effect needs to be immediate, othertimes a delay between the two is more suggestive. Changing between different modes of operation (e.g. hard-to-control and easy-to-control) within the one performance can also be more intriguing than maintaining a single set of relationships.

(4) Similarly, it was often valuable to deploy the stage space so as to have different areas of 'interactive potential'. For example, areas outside the range of tracking, or where unreliable results were returned by the technology, could be exploited for their dramatic properties just as much as more 'focal' regions where the tracking worked reliably. Indeed, having a differentiated performance space (with different 'sensitivities' or so that different things happen in different locales) was often more conducive to engaging performance than an 'isotropic' interaction environment would have been.

(5) The issue of how data from performers is mapped to algorithmically mediated effects is at the core of much of the above as well as being a significant issue in its own right. In an electronic arena, multiple mappings need to be coordinated both simultaneously and as an event unfolds over time. We have identified the strong need for an identifiable and coherently separable software component to support the mapping and remapping of data from participants. In this workpackage, the refinement of such technology is a major technical development task to be undertaken at the ZKM in the remainder of eRENA. The initial design and the current status of implementation work is described next in this deliverable.

APPENDIX

Workshop Participants

Ariane Andereggen, Switzerland
Cyril Bourgois, France
Susan Kozel, Ireland
Ramon Rivero, Mexico/ New Zealand
Gisèle Vienne, Austria
Kirk Woolford, United States

Workshop director : Sally Jane Norman

Workshop documentation : Jacques Sirot
(all photographs are framegrabs from video documentation)

Workshop team at the International Institute of Puppetry

Directress : Margareta Niculescu
Administration : Rodolph Di Sabatino
Accommodation and logistics : Brigitte Behr
Technical coordination : François Charneux

Workshop team at the Zentrum für Kunst und Medientechnologie

Institute for Visual Media

Director : Jeffrey Shaw
Technical coordination : Manfred Hauffen
Administration : Silke Sutter, Jan Gerigk
Workshop assistants : Simone van gen Hassend, Marie Blunck
Programming and Animation Team :
Sabine Hirtes
Michael Hoch
Bernd Lintermann
Detlev Schwabe
Andreas Schiffler

ZKM Medientheater

Technical stage management : Hartmut Bruckner
Lighting designers : Tommy Weimer, Werner Wenzel
Lighting assistant : Marie Blunck
Video beam director : Thomas Poser
Video Crew : Alex Ekonomidis, Esther Schlicht

Chapter Two
A General Framework for Transforming, Mapping and Choreographing User Interface Data for Performance Purposes in Electronic Arenas

Bernd Lintermann
Zentrum für Kunst und Medientechnologie (ZKM), Karlsruhe, Germany

2.1. Introduction

In the context of performances within an electronic arena, the generation of adequate control data for driving the graphics as well as the sound plays an important role. Since performances usually have a highly dynamic structure and are based on a certain choreography, the flexibility of data interpretation and mapping onto the graphics and sound parameters is crucial.

The common approach is to read raw user interface data, such as that generated by data gloves, magnetic or camera based tracking systems, into the graphics system of choice and to process the raw data within this system with the internally provided mechanisms. The graphics systems have to provide simple mapping mechanisms, a graphics language or a plugin mechanism. In the worst case, a programmer will be required to change the software source code.

Based on the experiences in the workshop *Real Gestures, Virtual Environments,* we decided to develop a toolkit which separates the processing of raw interface data into meaningful high level control data from the graphics system itself. The goal is to make the mapping task independent from the graphics generation software and thus achieve greater flexibility. The mappings should be adjustable with regard to the specific requirements of a certain performance and reusable in performances that rely on different graphics systems. This approach conforms with the software development principle of breaking down a problem into smaller entities and implementing solutions in encapsulated modules. It is planned to test this approach in Year 3 of eRENA in an experimental performance in co-operation with a dancer from the Frankfurt Ballet.

This document first explains the motivation behind the development of the software, next gives a description of the goals to be reached and then compares the proposed data pipeline to actual practice and discusses the system architecture decisions. Finally, system implementation issues are addressed.

2.2. Motivation

In the workshop *Real Gestures, Virtual Environments* held in August 1998 at the ZKM, the software Xfrog, developed at the ZKM, was used as graphics software for five pieces: an interactive growing tree, a moving abstract flower-like organic object, a simple human stick figure, an abstract calligraphic animation and a virtual fabric deformed by several performers (see Chapter 1). In all cases, one or more performers wore sensors of a Polhemus™ magnetic tracker device that was the only input device besides the computer keyboard and mouse. It transpired that in only two of the five pieces could the same mapping technique be reused. Besides the already

developed mappings, including a reconstruction of the human skeleton based on sensor values for extracting posture information, three new mapping techniques, in addition to adjustments and extensions of the existing mappings, had to be implemented.

Though the new implemented techniques are now part of the Xfrog software and can be used for other pieces using the Polhemus tracker device, it has over-stretched the software in terms of code complexity as well as in the number of offered user interface components. It also slowed the development of the pieces themselves, since sometimes the performers had to wait up to one day while the required techniques were programmed. The software also influenced the workflow on stage in a way unfamiliar to the performers. It would be desirable to compose mappings on the stage interactively and involve the performers in the process of testing.

One piece, the human stick figure, required a state change of the graphics system during the performance. Triggered by a certain event at the beginning, a single figure should unfold to a group of figures. Instead of giving the performer control over this event, a separate person had to press a key on the computer keyboard. A change in the graphics generation, as mentioned above, had to be triggered by a person different from the performer. This kind of change in data interpretation during the performance, which—using stage terminology—can be regarded as choreography of the data, is nearly impossible in all commercial systems.

Mapping problems slowed down performance development and interfered with the workflow in all performances. Therefore, we decided to explore an approach which in terms of software separates the mapping process from the graphics application and allows the interactive composition of mappings on the stage as far as possible. Though one can expect every technically complex performance to require additional software development, we hope to achieve a solution better adjusted to the work in a stage situation.

2.3. Goals

The software to be developed is intended to be used in the context of performances involving computer generated real-time graphics. It is a separate software module that links the interface hardware to the graphics generation software. It maps raw hardware device data to high-level application control data.

It should be interactive to allow non-programmers to use it easily and to create, test and adjust mappings directly on the stage, supporting a feedback oriented workflow. In case a performance requires specialised mappings that cannot be created with the offered functionality, it should at least be possible to program these mappings and plug them into the system. These plug-ins should be reusable for other performances. The plug-in architecture should support developers to focus on the algorithms rather than on user interface issues.

The software should be capable of choreographing the data processed by means of changing the processing method for control values dependent on the performance state and of defining smooth transitions between control values during performance. The keywords for the goals to be reached are:

- Flexibility
- Usability
- Extensibility

Since the software explicitly should support reusability of modules, it is not intended to focus on a specific type of performance or user interface hardware. The system architecture should be designed to support mappings in general. The term mapping is defined as the computation of control values depending on incoming interface data and the actual System State.

The software will be developed and tested in an experimental performance with a dancer from the Frankfurt Ballet. The Polhemus UltraTrak/StarTrak is planned to be the interface hardware.

2.4. Data Pipeline

Many graphics applications can already process incoming data and map it onto arbitrary graphical parameters like transformations, deformations or colour changes. MayaTM and SoftimageTM, for example, can map incoming data onto any kind of node attributes. Maya additionally offers a high level programming language (MEL, Maya Embedded Language), syntactically similar to UNIX shell scripts. The interface data is read by so-called device drivers that have to be rewritten for distinct hardware interfaces. MAXTM, a sound system for the Mac developed at IRCAM and commercialised by Opcode Systems, has a mapping mechanism customising splines and other features for transforming incoming data to drive parameters of a sound application. Whereas simple mappings are easy to create using the standard user interfaces, more complex mappings require either high level programming skills or an in-depth knowledge of the relevant software system. Usually the architecture for a system driven by an external user interface follows the scheme in Figure 2.1:

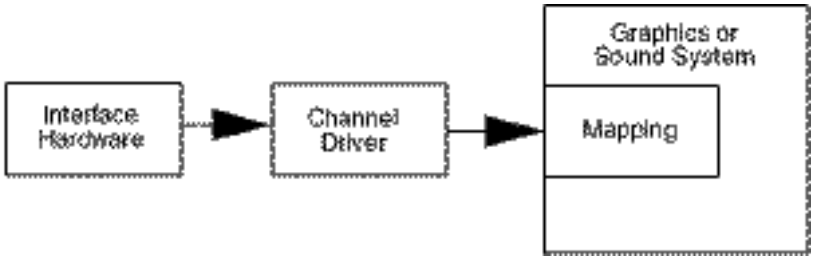

Figure 2.1: Standard data pipeline with external interface hardware

The channel driver is the software interface for the application to the hardware. Usually, it simply translates the incoming data into the data format of the application. The processing and mapping of the control data resides on the application. The following scheme illustrates this approach:

Figure 2.2: Standard system architecture with external interface hardware

Since in performances—in contrast to film production, for which most graphics applications are designed—the same graphics application must continuously generate changing graphics, make state transitions and vary the interpretation of the incoming data in time, the development of the mapping can become a complex task. The proposed architecture therefore introduces an additional module in the data pipeline, which pre-processes the incoming data and generates

graphics control data on a high level. The mapping residing at the application stays simple and ideally, just connects the generated control values to graphics parameters.

The idea is illustrated below in a variation of Figure 2.1:

Figure 2.33: Proposed system architecture with external interface hardware

The architecture of the system data pipeline shown in the first figure just introduces a new module that is illustrated by the figure below:

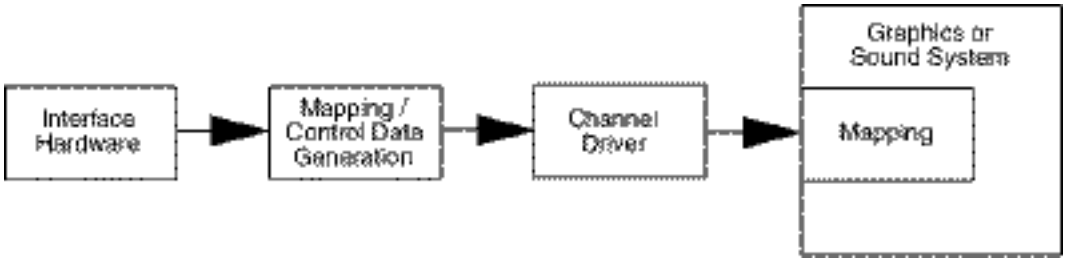

Figure 2.4: Proposed data pipeline with external interface hardware

2.5. System Discussion

2.5.1 Usability

Usability in a performance context addresses on the one hand the speed at which new mapping configurations can be created and tested on stage and on the other hand the technical skills that the user needs to make the necessary changes. Obviously the system has to be highly interactive: no programming should be involved on the stage and changes should have immediate feedback without the necessity for recompiling code. A graphical user interface is desirable. All parameters should be edited interactively. Also, it should be possible to define the interdependencies between mappings.

2.5.2 Flexibility

In principle, a programming language allows the highest flexibility in the creation of mappings, but in respect to usability, we concentrate on representations that can be edited graphically. The most common graphical representations that come closest to the computational power of programming languages are graphs. There are several interpretations of graphs. Directed acyclic graphs are used for representing rule systems. Other ones define the control/data flow between computational units. LogicTM, EddiTM, MayaTM and MAXTM are examples of systems in which networks generate sound, images, 3D graphics or MIDI data.

The computational units or nodes encapsulate one problem solution and generate output data from input data. The whole problem is broken down into smaller problems that exchange the results of computations. The commercial systems mentioned above differ in the type of data with which they are dealing, as well as in the evaluation strategy of the network.

Due to its flexibility and interactivity, the mapping toolkit's software architecture is based on the network approach. Here, nodes are computational units that map input data onto output data in different ways. A complex mapping is constructed, creating a network of simple mappings. The mapping toolkit should provide a basic set of nodes for e.g. arithmetical operations, expressions, splines, tables and deal with data types often used in graphical applications such as geometric transformations and vectors. Since the toolkit processes data in a pipeline there must be nodes that read hardware device data and other ones that communicate the computed control values to the graphics application. A database concept should be integrated, in order to have access to external, precomputed data.

Since nearly all graphics applications are using connected nodes for the manipulation of at least the scene graph, one can expect users to be familiar with the concept of connected nodes.

2.5.3 Extensibility

Of course, it is impossible to provide all kinds of mappings that might be required in any performance. Even if a complex mapping could be constructed out of the default mapping mechanisms, it would make sense to reprogram them for speed reasons or even only to have a more compact user interface. For several years, commercial software companies have enabled third party developers to extend the basic functionality of their products by plugging in external code and thereby to add value. Technically, this achieved by linking dynamic shared objects at runtime.

The success of these plug-in architectures has enabled us to decide that the plug-in mechanism should be the main means of extending the software. The plug-in architecture should allow the developer to focus on the topic of their interest, the algorithm, which means that as little code as possible should be written in the implementation and the user interface should be created automatically. Previously developed code should be as reusable as possible.

Since the chosen software architecture is a network of nodes with connected parameters, it should be possible to write new nodes as well as to define new parameter types that can be used by nodes. For parameters with no suitable existing editing methods, it should be possible to implement graphical editors.

2.6. System Architecture Description

The following paragraphs describe the actual system architecture and introduce the terminology used.

2.6.1 Network

Users should be able to build mappings needed for their performance out of predefined computation units. Units are designed in a way that they implement problem solutions common to most mapping problems, e.g. the interpolation of different values, rescaling/transposing of

data, and mapping using expressions or splines. It should, for example, be possible to implement flocking behaviour, chaotic mappings and others in these computational units.

The basic computation units visible to the user are *Nodes*. Each node has a parameter set called *Attributes*. The attributes of a node can be connected to attributes of other nodes.

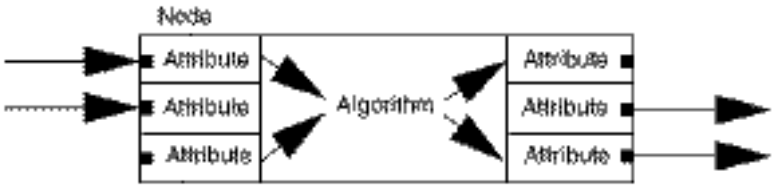

Figure 2.5: Diagram of a node with attributes

A *Connection* is a directed link between two attributes of different nodes. A connection invokes the attribute value of the destination attribute to be overwritten by the attribute value of the source attribute. The connectivity of nodes in a *Network* determines the flow of data between the nodes. There are special *Device Nodes* which communicate with the interface hardware and the graphics application.

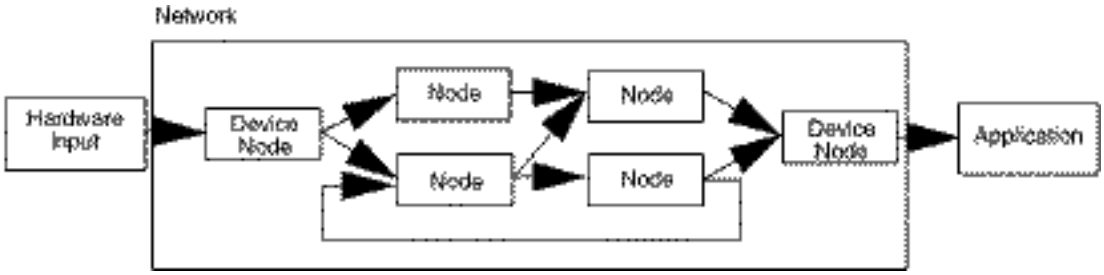

Figure 2.6: Diagram of the data flow of a network connected with the hardware device and the application

2.6.1.1 Evaluation Strategy

A dependency oriented evaluation strategy, such as the one on which Maya is based, is transparent in terms of time behaviour, because the system takes care of the correct evaluation order of the nodes. A node is evaluated only if all nodes providing data for it have been evaluated. The network behaves correctly if the data flow and the dependencies are defined correctly.

In control flow oriented models, such as the one on which MAX is based, the user has to keep track of the evaluation order of the nodes, which often becomes a difficult task with the increasing complexity of the network.

Whereas the dependency oriented model is easier to understand, it lacks the capabilities of the control flow model, such as selective evaluation of nodes, data base accesses and multiple evaluation of nodes. Due to its transparency, the mapping toolkit is based on the data flow model, but for increasing flexibility there are additional control flow features integrated.

Since the evaluation strategy is crucial for the network functionality, Appendix A to this chapter gives a more detailed description based on examples.

2.6.2 Plugin Architecture

A plugin architecture allows the implementation of computation units which solve very specialised mappings needed in special cases. The system architecture allows to write custom nodes easily. These nodes appear automatically in the graphical user interface and there is an automatic user interface builder implemented for editing the attributes of a node. Once an attribute is created and certain flags are set (e.g. if it is connectable to other attributes, writable, readable etc.), the user interface for the node is generated automatically.

Because attributes and nodes are implemented in C++, custom nodes and attributes can be derived from existing ones while inheriting their functionality. Thus, programmers can refine the functionality of existing nodes if they encounter a special case that the standard nodes cannot handle. Appendix B to this chapter gives an example code for the definition of a node that interpolates between two values.

2.6.3 Software Architecture

Beside the conceptual issues discussed above an extensible system requires a clear software architecture. Plug-in programmers should be able to write nodes without worrying about user interface issues. For usability there should be an effective Undo feature. These considerations led to an internal software architecture with separated software layers, a Network Layer which implements the network functionality, a Manipulation Layer which keeps track of users' actions and provides undo functionality, and a User Interface Layer which manages user interface elements like dialogs, sliders etc.

Figure 7 illustrates this structure:

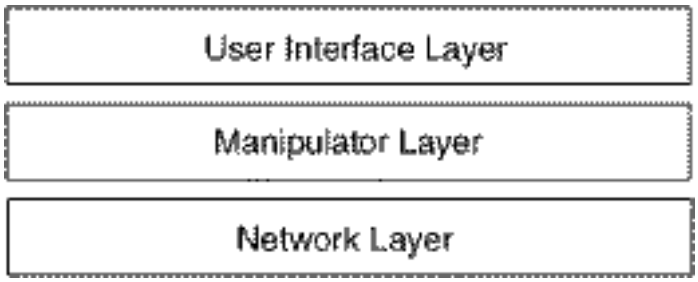

Figure 2.7: Software architecture layers

2.6.3.1 Network Layer

The Network Layer provides functionality for creating nodes and attributes and manages all the evaluation of the network. Custom nodes can be created and introduced to the system by a simple registering mechanism. Also, new attribute types can be defined that inherit the functionality of existing attributes. After an attribute is registered, it is usable by any node written by the current or any other plugins. A registered node appears automatically in the user interface and can be used like any other node. Interdependencies of plug-ins are resolved automatically.

The class **DeviceNode** defines two virtual methods—**preCompute** and **postCompute**—which are called before and after evaluation of the network. Additionally, they provide an interface for spawning parallel processes and provide functionality for communication between the node and

the spawned processes. These device nodes can be used for reading and writing data to and from blocking hardware devices.

2.6.3.2 Manipulation Layer

The Manipulation Layer provides user interface objects with manipulation functionality to the network. Manipulators represent and change the state, e.g. the connectivity, of the network. They provide methods for displaying themselves and should be able to undo an action as well as they are check interdependencies of actions. As an example of a manipulator, the connection manipulator represents a connection as a sequence of lines. It has methods for selecting, moving, displaying the line segments and for creation and removal of a connection. If the user creates a connection via user interface, he or she creates a connection manipulator object in the manipulator layer which then creates the connection in the network layer. Since manipulators are representations of network state changes, they are used by the application history to implement infinite undo and redo functionality.

Just as with nodes and attributes, a programmer can define new manipulators and register them to the system.

2.6.3.3 User Interface Layer

The user interface for node attributes is created automatically from the attribute's definition. Since the user can define new attributes, he or she may also wish to define the user interface for editing that attribute. Like nodes, attributes and manipulators, new editors can be assigned to new attributes as well as existing ones with a registering mechanism. Users can even redefine default editors for the basic attributes like floating point or string attributes.

2.7. Implementation

2.7.1 Attributes

Attributes are defined as C++ Classes. Several virtual methods allow the redefinition of the attribute behaviour, e.g. reading and storing of data to a file or the setting of values. Certain macros define automatically the necessary existence operators, like creator, copy and the registering methods. An accept method checks for type consistency of connections.

All attributes are derived out of four atomic attribute types: *Int, Float, String, Pointer* and *Compound*. Int and Float implement the basic numeric types. The String type is a sequence of characters (letters, numbers or symbols). Compound attributes are containers of attributes. They are used to create arrays or hierarchical attributes. Pointer attributes allow a programmer to attach any kind of data, for example image or sound data, to an attribute.

Dynamic attributes support data type abstraction for operations which might be applicable to different data types. For example, subtraction, addition or the inverse are applicable to floating point values and vectors, as well as geometrical transformations. The Dynamic attribute allows only one of a set of declared data types to connect, and the application can process the required operation type selectively. Table 2.1 lists the already implemented attribute types:

Attribute Type	Description	User Interface	Inherits
Float	Floating point value	Slider	Atomic
Int	Integer value	Slider	Atomic
Boolean	Boolean value	Check Box	Int
Option	Choice between offers	Option Box	Int
String	String of letters	Text Edit	Atomic
Compound	Container Attribute	browsable	Atomic
Array	Array of arbitrary attributes	browsable	Compound
FloatArray	Array of floating point values	browsable, Scale sliders	Array
Vector2D/3D/4D	Array of 2/3/4 floating point values	browsable, Scale sliders	FloatArray
VectorArray	Array of Array of floating point values	browsable, Scale sliders	Array
Spline	VectorArray with Spline functionality	Spline Editor, Preview	VectorArray
Transform	3D Translation/Rotation/Scale	browsable, Scale sliders	Compound
Range	defines a Range by two float values	two Scale slider	FloatArray
Image	Pointer to an Image	File Name Edit, browsable	Pointer
Dynamic	Dynamic	-	Compound

Table 2.1: Implemented attributes

2.7.2 Nodes

Nodes are defined as C++ Classes. Similar to attributes, node classes are derived from existing classes inheriting their behaviour. The system offers two basic node types: **Node** and **DeviceNode**. Any node is derived from Node as the base class.

The class DeviceNode has additional virtual methods for communication and methods for initialization and the handling of asynchronous processes. The methods **preCompute** and **postCompute**, which are called before and after the evaluation of the network, are intended to read data from a device before and writing control values to the graphics application after each frame.

The following nodes are implemented for testing certain concepts. The number of mappings still has to be expanded. Besides the discussed mapping nodes, which read from an external hardware device, there are two nodes (the Time node and the Mutation node) which generate values based on constraint randomisation and on the actual system time or frame.

Blend:
Input: two values, blendfactor
Description: computes the weighted sum between two given values
Output: weighted sum

Spline:
Input: array of 2D control points, argument
Description: linear and bezier interpolation of a graphical edited curve of 2D control points

Output: interpolated value

Expression:
Input: expression, argument
Description: evaluation of a user definable function
Output: function result

Time:
Input: stop/go, length
Description: generates a value depending on the system time or frame, stop and go, loops, spline and functional mappings integrated.
Output: time value

Mutation:
Input: speed, preferred position, amplitude, acceleration
Description: generates a random smooth changing value within the given constraints. The value changes with the given speed around a preferred position with a maximum amplitude.
Output: random value

Select:
Input: float value, selection value
Description: selective evaluation of connected nodes, the output attribute is an FloatArray. Depending on the incoming selection value, one element out of the array is chosen and the connected node is forced to be evaluated with an additional input value.
Output: array of floats

Print:
Input: float|vector|transformation
Description: prints the value of an incoming float, vector or transformation to a log window.
Output: input value

Inverse:
Input: float|vector|transformation
Description: inverts the incoming float, vector or geometrical transformation.
Output: inverted input value

2.7.3 User Interface

2.7.3.1 Network

Figure 2.8 shows the user interface main window of the toolkit in the actual implementation. A simple network of nodes illustrates the different components.

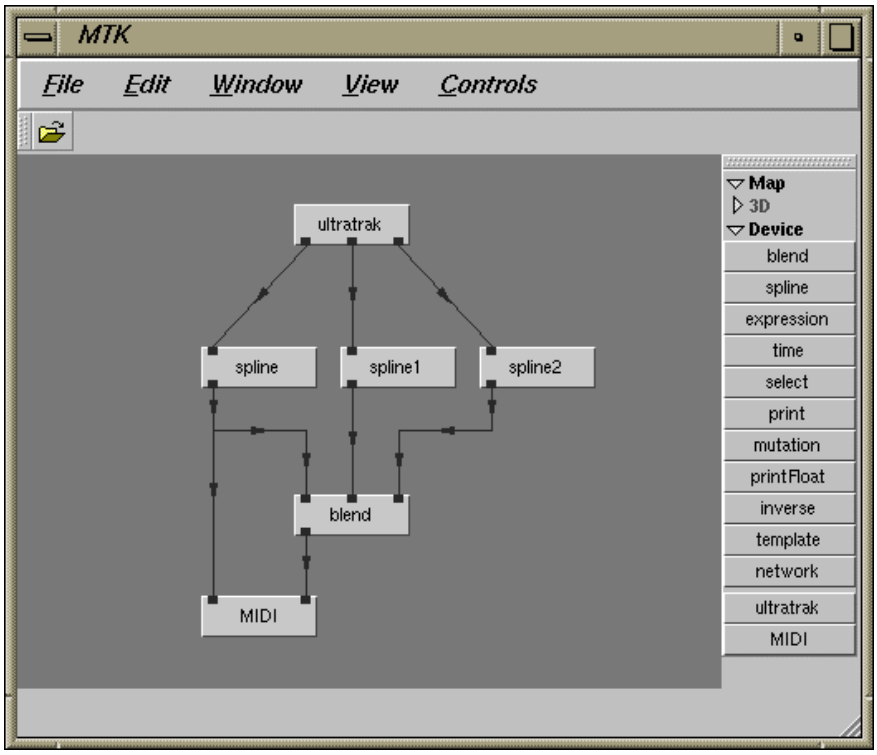

Figure 2.8: Application main window

For quick access, Input and output attributes have small squares in top (Input) and at the bottom (Output) of the nodes.

The **Ultratrak** and the **MIDI** nodes in the example are Device Nodes which, respectively, read data from the Ultratrak magnetic tracker device and write data to the MIDI device.

The **Spline** nodes map sensor data via splines. The output of two spline nodes is blended with the blend factor coming from the third spline.

In the toolbar to the right of the window, all available node types are offered, browsable through user definable categories (here Map, 3D, Device).

Figure 2.9 shows an example of the parameter dialog for a node, with some of the implemented attribute types like Float (Output), String (File Name), Bool (Switch), Option (Protocol), Vector3D (Translation), Compound (Blend consisting of a Float (Alpha) and a Vector2D (Limits)) and VectorArray (Control Points).

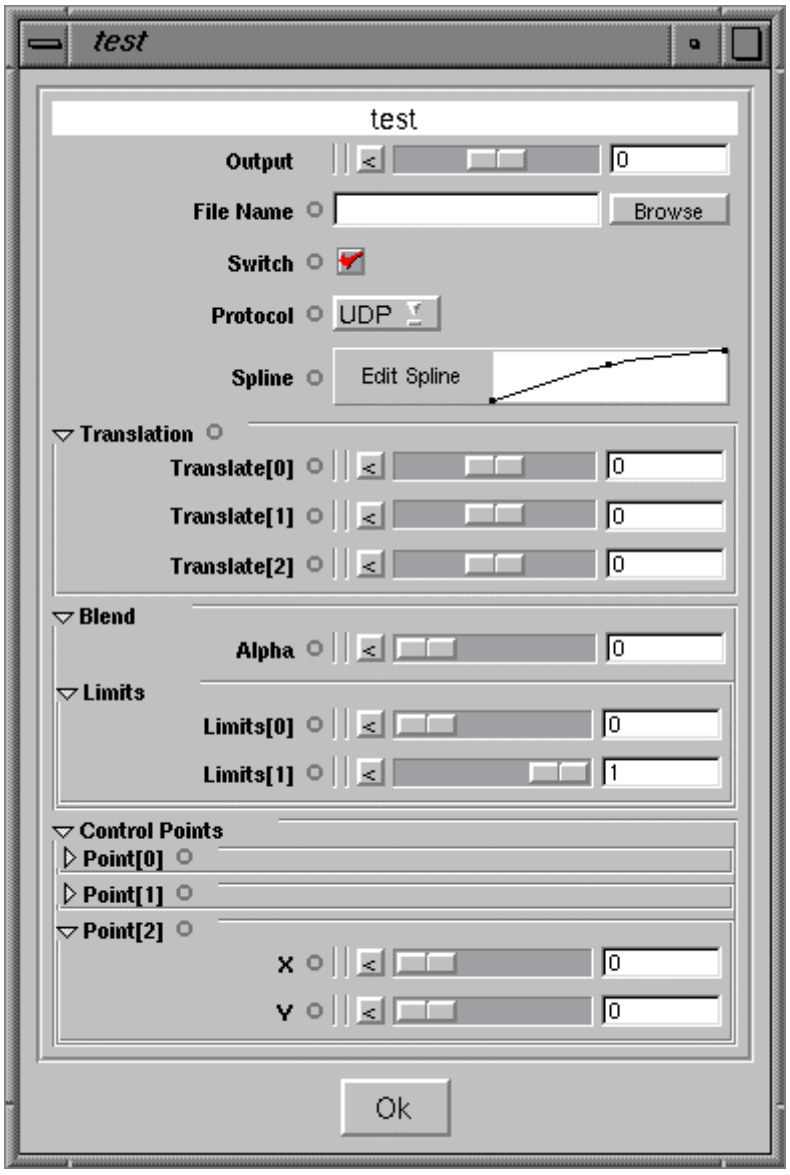

Figure 2.9: Node parameter dialog

Attributes can be connected firstly by using shortcuts, visible as small squares in the node icons (see figure above), secondly by dragging them from the parameter dialogs and thirdly with the connection editor, which allows the browsing of the attributes of two nodes which are to be connected hierarchically.

The spline button shows a preview to the spline, which allows a rough edit without opening the spline dialog.

2.7.3.2 Sub-Networks

Since a network consisting of a lot of nodes can become visually complex, there is a need for some visual structure. A Network Node allows the user to edit a subnetwork in a separated window.

In Figure 2.10 the main network is defined in the back window. A network node in the middle interfaces the subnet with three blend nodes. The connected attributes are accessible in the subnet via the interface components **Input** and **Output** at the top and the bottom of the subnet window. Input and output attributes are defined by dragging them from the source node with the mouse onto the Input or Output components.

Network nodes behave like any other node in that they can be seen as visual abstractions of mappings. It is intended to make the subnetworks selectively storable and reloadable, to allow the creation of libraries of high level mappings.

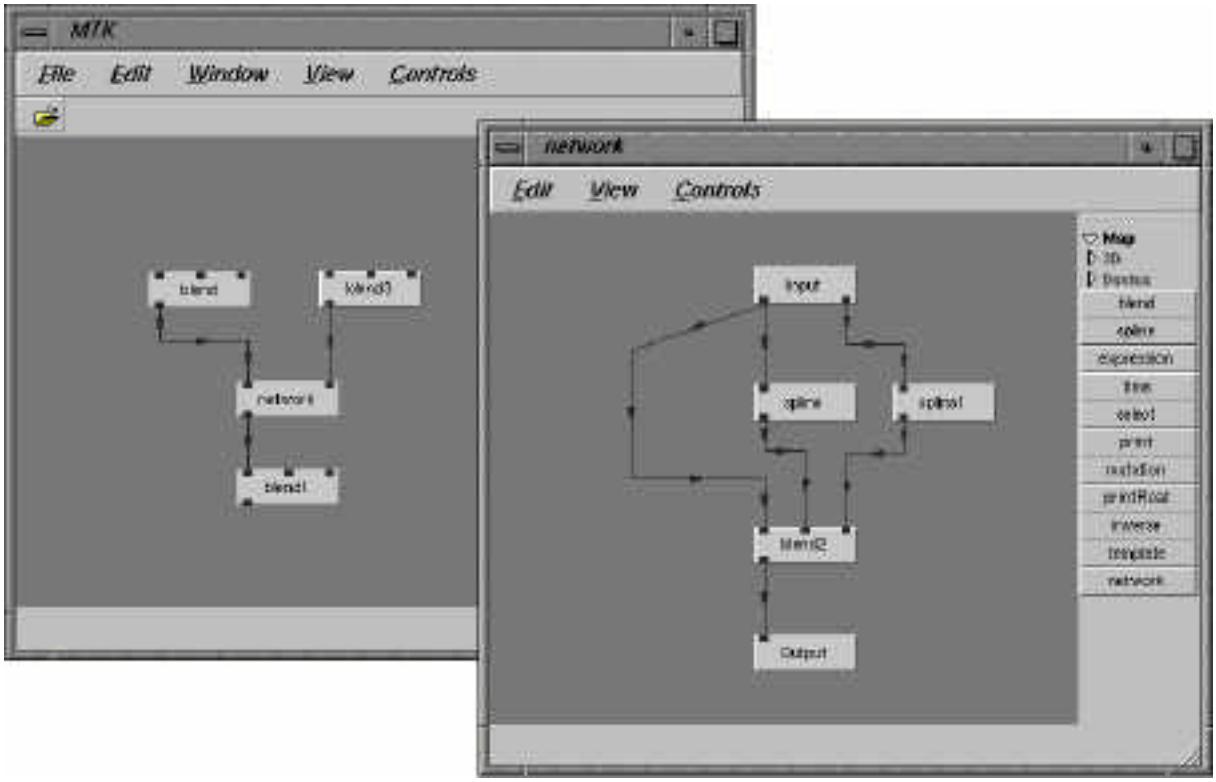

Figure 2.10: Main window and subnetwork window in the front

2.7.3.3 Usability Features

Besides the general system design, the usability of a system usually depends on at first sight small, yet crucial, details. Since usability is one of the main goals, there follows a list of implemented features without going into detail.

The system supports single and multiple selection. Selected nodes and connections can be moved, deleted, and automatically rearranged. A node is created by dragging it from the toolbar into the edit area. All actions can be un- and re-done with an infinite undo history.

Nodes can be activated and deactivated. Deactivated nodes are not evaluated or dependency checked. This prevents evaluation of subnetworks if not needed. For reducing visual complexity, nodes can be hidden. In that case, they are not displayed in the network.

Nodes can be copied as whole objects or as a reference. A reference node is intended to be used if a node is needed at several locations in the network to avoid long distance connections.

Attributes can be connected firstly by using shortcuts visible as small squares in the node icons, secondly by dragging them from the parameter dialogs and thirdly with the connection editor. The connection editor browses the attributes of two nodes to be connected hierarchically. The appearance of the connection editor is similar to the one in Maya.

An attribute browser inspects and sets certain properties of a node's attributes interactively, e.g. the visibility in the parameter dialog or the attribute name, if a shortcut for the attribute is to be shown or if the attribute is to be dependency checked.

Moving the mouse over an attribute or a connection gives a brief description.

A Plugin Manager allows the user to load plug-ins on the fly. Plug-ins can be automatically loaded when the system starts up.

There is a simple remapping implemented on floating point attributes. The absolute value of a connected attribute is not read by the application, but its relative position in the parameter limits defined in the user interface is mapped onto the limits of the parameter it is connected with. This is useful for algorithms that generate values within the normalized limits 0 and 1. The actual value the node reads is directly edited in the parameter editor of the node without introducing a new node that scales and shifts the value to the required parameter space.

The network visual size is smoothly scalable.

2.8. Summary

Based on the experiences in the workshop *Real Gestures, Virtual Environments,* we decided to develop a piece of software specifically for mappings for performance purposes. It introduces a new module in the data pipeline from the hardware device to the graphics or sound application.

The separation of the mappings from the application makes the mapping independent from the capabilities of the application and allows us to design a software architecture which supports the mapping creations in terms for flexibility, usability and extensibility.

The system architecture is a network based system with a dependency based evaluation strategy, which offers additional possibilities for selectively evaluating nodes.

The implementation offers a completely interactive user interface, a visual representation and manipulation of the network and the node attributes for quick creation and adjustment of mappings on the stage. Subnetworks allow the visual separation of a complex network into smaller modules.

The functionality of the system can be extended by a programmer via a software plug-in mechanism. Very special mappings and data types can be developed and loaded into the system on demand. This can be utilized for building libraries of mappings.

A small number of nodes and attributes supporting standard mappings are already implemented. More mappings have to be implemented on the base of a planned real performance.

The focus until now has been on the design and implementation of a clean software architecture and usability features rather than on the mappings themselves. In Year 3 of eRENA, work will be conducted to explore the kinds of mappings that are useful in different performance contexts. Experience in *Real Gestures, Virtual Environments* already gives some clues to this, as do the findings of other partners working in real-time performance settings for electronic arenas. It is

our contention that the nature of the mappings between input data and control parameters is essential to the quality of interactive experience when engaged with the complex computer graphical and sonic material characteristic of an electronic arena. Understanding which kinds of mapping are intelligibly performable in which contexts is a clear future topic to be investigated of cross-workpackage relevance (especially between Workpackages 4 and 6) and of cross-partner interest (especially between the ZKM and KTH).

Appendix A

Evaluation Strategy

The evaluation of the network is frame based. Each node has a time stamp to determine if the node has been already evaluated in the current frame. If a node is to be evaluated in a frame, the system sets the current time stamp and then tests if all nodes which provide data for this node have the current time stamp. If one of these nodes has no valid time stamp, its time stamp is actualized and then evaluated with the same dependency checking first. Finally all successors of an evaluated node are checked. The algorithm for the evaluation looks like that:

```
Evaluate( Node )
{
   TimeStamp = Current Time Stamp;

   for all nodes which point to this Node
      if  ( node.TimeStamp < Current Time Stamp )
         Evaluate( node );

   Compute();

  for all nodes which Node points to
      if  ( node.TimeStamp < Current Time Stamp )
         Evaluate( node );
}
```

An example should clarify this procedure. Let us assume that Node3 in Figure 2.11 is to be evaluated. First, its time stamp is set and then Node2 is checked. Since Node2 has no current time stamp, it is evaluated. Its time stamp is actualized and then Node2 checks if Node1 has to be evaluated. The time stamp of Node3 is set and then, since it is independent from any input, the algorithm of Node1 is invoked. After evaluation of Node1, Node2 checks Node3. Because Node3 has the actual time stamp, Node2's algorithm is invoked and the control goes back to Node3. Its algorithm is invoked and then its successor, Node4, is checked.

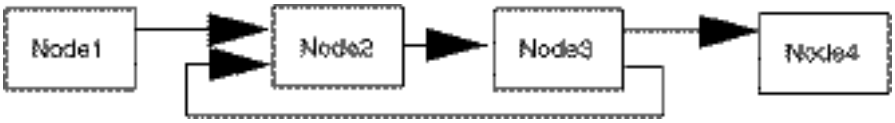

Figure 2.11: Data flow between nodes

Breaking the pure dependency oriented strategy, the algorithm of a node can force all nodes connected to a certain attribute to be evaluated. In this way, nodes can be evaluated several times

during one frame. A connection which connects such an attribute is marked with a double arrow. The evaluation is forced by decreasing the time stamp and calling the evaluation procedure for all nodes connected to the attribute.

In Figure 2.12, during execution of its algorithm Node2 could evaluate Node3 several times with different input values. If Node3 stores precomputed values and updates the output attribute according to the incoming value, this is equivalent to a database query.

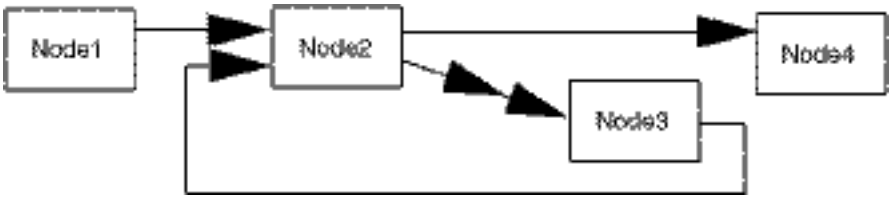

Figure 2.12: Forced evaluation of a node

Figure 2.13 shows the setup of a selective node evaluation. Node2 could force either Node3 or Node4 to be evaluated depending on the incoming value from Node1. If additionally the outputs of Node3 and Node4 are blended with a blend factor depending on Node1, this models a smooth transition between mappings.

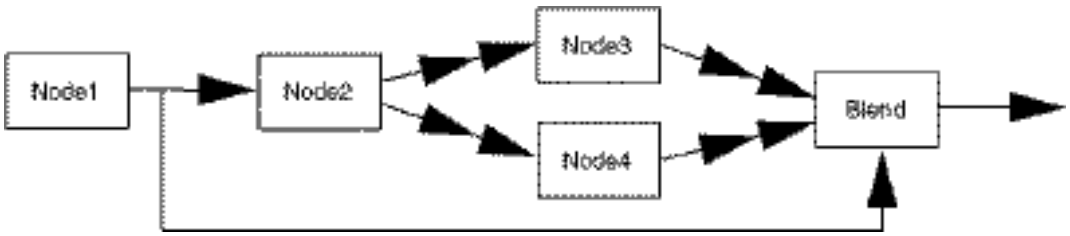

Figure 2.13: Selective evaluation of two nodes

Appendix B

Example source code for a node definition

```
/*************************************************************************
*** FILE: mtBlendNode.h***
*************************************************************************/

#ifndef _MT_BLENDNODE_H_
#define _MT_BLENDNODE_H_

#include "mtNode.h"

class mtBlendNode : public mtNode
{
  mtFloatAttributePtr      m_Output;
  mtFloatAttributePtr      m_Alpha;
  mtVector2DAttributePtr   m_Limits;

public:
  mtBlendNode();
  ~mtBlendNode();

    void           compute();

  MT_NODE(mtBlendNode)
};

#endif

/*************************************************************************
*** FILE: mtBlendNode.cc***
*************************************************************************/

#include <stdio.h>
#include "mtGlobal.h"
#include "mtBlendNode.h"

/*************************************************************************
*** MACRO DEFINING CREATOR, COPY, REGISTER... METHODS
*************************************************************************/

MT_NODE_HANDLER(mtBlendNode,"blend")

/*************************************************************************
*** mtBlendNode Creation/Destruction
*************************************************************************/

   mtBlendNode::mtBlendNode(){
  m_Alpha  = (mtFloatAttribute    *) createAttribute( mt::mtFloat,
"alpha",  "a" );
  m_Limits = (mtVector2DAttribute *) createAttribute( mt::mtVector2D,
"limits", "l" );
  m_Output = (mtFloatAttribute    *) createAttribute( mt::mtFloat,
"output", "o" );
```

```
   m_Limits->setElementShortCut( true );
  m_Limits->setReadable( false );
  m_Limits->element(0)->setFloat( 0.0 );
  m_Limits->element(1)->setFloat( 1.0 );
  m_Output->setWritable ( false );
  m_Output->setShortCut  ( true  );
   m_Alpha->setReadable( false );
  m_Alpha->setShortCut( true );
}

mtBlendNode::~mtBlendNode()

{}

/*************************************************************************
*** mtBlendNode compute Method called during evaluation
*************************************************************************/

void
mtBlendNode::compute()

{
  float alpha  = m_Alpha->getFloat();
  float l0     = m_Limits->element(0)->getFloat();
  float l1     = m_Limits->element(1)->getFloat();
  float result = (1-alpha)* l0 + alpha * l1;

  m_Output->setFloat( result );
}
```

Chapter Three
Event Management in Electronic Arenas by Visualising Participant Activity and Supporting Virtual Camera Deployment

John Bowers, Kai-Mikael Jää-Aro, Sten-Olof Hellström and Malin Carlzon
Royal Institute of Technology (KTH), Stockholm, Sweden

3.1 Introduction

This chapter introduces a major strand of work being conducted in Workpackage 4 devoted to the support of event management in electronic arenas. Chapter 4 following further develops the themes introduced here and Chapter 5 culminates in a demonstration of a prototype production support 'suite' which presents the ideas introduced here in a mixed reality environment intended for use by production personnel concerned with managing events in electronic arenas.

The work over these three chapters has a character that has been strongly influenced by the ethnographic analyses of production work which have been carried out in Year 2 of eRENA. In particular, social scientific study of the production of the inhabited television demonstrator *Out Of This World* has been rich in its yield of suggestions for future requirements of electronic arenas and how to respond to these. This demonstrator and study of the behind-the-scenes production work is reported in Deliverable D7a.1. In particular, Chapter 7 of that deliverable makes a number of proposals for future work. What has followed in Workpackage 4 is our response to these proposals.

To be more concrete, we have devoted ourselves to developing techniques for enhancing the deployment of virtual cameras in an electronic arena, for managing their relations, and for enabling production staff to search out the action in a potentially mass-participation environment. In each of these respects, we are providing tools which attend to specific needs expressed by the director of *OOTW* while doing so with a generality of approach that will enable, in principle, further resources to be at hand for producers of any event in an electronic arena. In the Preface to this deliverable, we emphasise how the most characteristic instantiations of the electronic arena concept will involve mass-participation in virtual and mixed reality environments rich in inter-media sources. It is management of scenarios of this sort that we are most concerned to support here.

To look ahead, we propose to do this by presenting production staff with real-time visualisations of participant position, orientation and activity while an event in an electronic arena unfolds. These visualisations can be interacted with so as to deploy virtual cameras. We propose a number of algorithms for the near-optimal initial deployment of cameras. We also propose a number of ideas for algorithmically controlled camera paths, including cameras which actively and autonomously seek out the action. However, we envisage such algorithmic techniques being used in concert with human manipulation and control. For example, algorithms might determine initial deployment but, following this, finer control may be achieved using a camera interface much like that developed for *OOTW* (see Deliverable D7a.1). Indeed, the work reported there is entirely complementary to the ideas here as we envisage complex mass-

participation electronic arenas requiring combinations of automatic and manual techniques to facilitate capturing an interesting selection of visual sources.

While the emphasis of the current chapter (and Chapter 5) is on the control and deployment of virtual cameras, our interest in managing electronic arenas is not confined to graphical material and the visual sense. An essential feature of electronic arenas is that they are media rich and enable the exploration of inter-media and inter-sensory relations. Thus, explorations of sound, and sound in relationship to vision, have been a feature of a number of the demonstrators and prototypes in the project (see, for example, the interactive soundscape that is part of *Murmuring Fields*, Deliverable D6.2, the use of video analysis to interact with sound discussed in Deliverable D6.3, the relations between sound and virtual world construction explored in the *Lightwork* performance presented in Year 1's Deliverable D2.2, and the practical relationship between sound and production work identified in the ethnography of *OOTW* reported in Deliverable D7a.1). Chapter 4 of this deliverable considers how we can 'sonify' participant activity in ways which might complement the current chapter's emphasis on the use of visualisations as a resource in event management. It may also be possible to extend our ideas for virtual camera control to notions of 'virtual microphones' which sample the audio at different locations within an electronic arena—a possibility we return to in Chapter 7. Indeed, just as participants to an electronic arena are confronted with a rich set of inter-related media, so we propose that production staff may profit from a 'rich sensorium' to facilitate their practical behind-the-scenes activity in managing events. The next few chapters are devoted to this possibility.

In addition to the influence empirical social scientific studies have had on our technical development work, we have been concerned to carefully and critically examine existing research into camera deployment, view control and the management of events in virtual environments. While the existing literature contains many interesting ideas and implementations, we have found it to be critically lacking in a number of respects. In particular, the specific features of electronic arenas that we wish to emphasise (real-time, potentially mass-participation, multiple media sources and so forth) are not centrally addressed in much existing work. Thus, a critical examination of existing research has enabled us to refine our design ideas while also realising the specificity of the electronic arena concept.

Of central importance in all this has been the development of what we *call activity-oriented camera control and deployment*. In many ways, this can be regarded as a particular instantiation of a general concept we would like to offer: *activity-oriented navigation*. Conventional virtual reality systems support *avatar-centred navigation* through the control of the position and orientation of the embodiment of the user. The camera control interface developed for inhabited television applications in eRENA supports *object-centred navigation* so that movements can be made in relationship to entities in the field of view (see Deliverable D7a.1, especially Chapter 3). In the current work, we are proposing a further paradigm of activity-oriented navigation whereby deployments in space can be influenced by activity within it. In Deliverable D6.3 we explore these ideas as a general purpose navigational aid. In the current chapter, we examine specific applications of this navigational paradigm for the deployment and control of virtual cameras.

Following this introduction, we present a critical examination of existing research into the control and deployment of virtual cameras in the management of virtual events. We put our analysis of this literature and the findings of our field research in eRENA together and draw out six emphases which we believe work on event management for electronic arenas should have.

We then go on to present our technical ideas for applications consistent with these emphases. This culminates in the presentation of an application, SVEA (Sonification and Visualisation for Electronic Arenas), which demonstrates our ideas in an implemented prototype. The chapter closes with a review of the current status and future directions of our work.

3.2. Critical Examination of Existing Research into Camera Deployment and Control in Virtual Environments

In this section, we offer a detailed review of what we believe to be the most important contributions to research into the deployment and control of cameras in virtual environments. Throughout, we appraise the claims made by authors against what we take to be the requirements of (specifically) electronic arenas and against our findings in empirical field research studying professional direction and camera work. This sets the stage for our own application development work which is described following this review. To facilitate reading, we structure our examination of the literature around two main headings—*Supporting Camera Deployment*, where we examine proposals for how the initial deployment of cameras can be effectively made, and *Supporting View Manipulation*, where we examine techniques for subsequently shaping what the camera picks up on in its field of view. In practice, though, researchers have often addressed both topics together or have covered both in the same paper. Nevertheless, this is a coherent logical separation (*where should cameras go?* versus *what should they do once there?*) and one which facilitates our presentation.

3.2.1. Supporting Camera Deployment

3.2.1.1. The Virtual Cinematographer

He, Cohen and Salesin (1996) describe The Virtual Cinematographer (VC), a system intended to support automatic real-time camera control and direction in a virtual environment. He et al.'s approach involves expressing 'idioms' of cinematography with the formal notation of finite state machines and defining cameras which calculate optimal shots given a specification of the state of action within the scene. Idioms would include two and three party talk. Cameras defined by He et al. include tracking, panning and following cameras, as well as those which give 'apex shots' showing the relationship between two parties. He et al. take a rather strong view over how the 'rules of cinematography' influence shot composition and direction. For example, on page 10 of their paper they state: "The rules of cinematography dictate that when the Line [the line of action between two actors, see our discussion in Deliverable D7a.1] remains constant, the camera should remain on the same side of the Line". Clearly, understanding 'rules' as operating in this fashion often enables the VC to geometrically determine 'optimum' camera locations and directions (though, on occasion, the VC will slightly adjust the position and orientation of actors if a better shot can be created thereby).

We imagine that He et al. are concerned to support direction for a particular kind of scenario: one which is scripted to a level of detail which enables the appropriate 'idiom' to be identified at any moment but where, nevertheless, actual camera deployments and edits can be computed in real-time. Their paper closes with a description of a simulated 'party' where autonomous actors freely join and leave multiple conversations within a room environment. In less constrained, live environments whether a participant has left or remains within a conversation, exactly when this occurs, and hence when a change to a different idiom (say three party to two party talk) occurs, may need to be the product of human judgement. Whether idioms could reliably recognise their

own applicability is a somewhat debatable affair. If transition between idioms need human judgement then more hybrid (human direction/autonomous deployment) scenarios probably need to be investigated.

A similar point can be made about the influence of 'rules of cinematography' on the real-life conduct of directors. Most can be 'broken' if appropriate dramatic effects can be gained thereby. Indeed, in the specific case of the supposed rule that the Line should not be crossed, we saw in *OOTW* the director working out the quite contrary view for the distributed action typical of the electronic arenas of inhabited TV. If the most promising examples of a rule-to-be-obeyed for He et al.'s approach can be broken or wisely ignored in an electronic arena, then we have reason to believe that an alternative orientation is required. Hence, in our work, we do not seek to formalise the supposedly implicit rules of cinematography.

3.2.1.2. Automatically Generated Illustrations

In a number of papers, Feiner and colleagues (Seligman and Feiner, 1991; Karp and Feiner, 1990; Feiner and McKeown, 1991) have discussed techniques for assembling sequences of illustrations of objects so as to communicate how such objects function and how complex tasks can be performed with them. The illustrations are optimised for such details as lighting and camera angle and position. It is clear that the intended use-scenario is a didactic one where instruction is give to potential users of complex machinery, say, or maintenance engineers and such like. Feiner and colleagues' techniques generate animations but are not intended for real-time usage, nor are they primarily designed for scenarios where social interaction forms the subject of shots.

It is possible that components of the systems developed by Feiner and colleagues could be of use in electronic arenas. For example, it may be appropriate to automatically compute hints as to how a scene should be composed, lit or shot. (For similar work automating the 'mise-en-scène' of animations, see Hoppe, Gatzky and Strothotte, 1995.)

As we have seen other authors do, Karp and Feiner (1990) also present orthodox cinema practices (e.g. continuity editing, the prohibition of crossing the line) rather uncritically and while the automation of such practices may be appropriate to instructional domains, as we have discussed already, there is no reason to encode them in systems supporting the live-action participatory artistic, entertainment and cultural events we envisage occurring in electronic arenas.

3.2.1.3. Variably Parameterisable Cameras

Mulder and van Wijk (1995) present a method for defining multiple views in 3D virtual environments. Their principal application focus is on complex simulations where the researcher can change, in real-time, the parameters of what is presented and receive immediate feedback on the results. This emphasis on real-time, interactive operation and multiple views they share with us. Mulder and van Wijk introduce the notion of a 'point-based parameterisable camera object' and discuss how such cameras could have their views influenced by direct manipulation, by data from the ongoing simulation, or by customised camera controls. In one presentation of the idea, small camera objects are rendered in the view of a complex environment and, when selected, 'control points' become visible on the camera. These can control the direction, orientation and other features of the camera view. In this scenario, the user is deploying and manipulating cameras and their views in a hybrid viewer/director/camera operator role. For our purposes, the

interest in Mulder and van Wijk's work lies in the utility of presenting cameras themselves in a view of the environment which a director could work with while allowing both direct and algorithmically mediated (e.g. parameterised by the ongoing simulation) manipulation of them.

3.2.2. Supporting View Manipulation

3.2.2.1. Manipulation Metaphors

Ware and Osborne (1990) systematically investigated three different metaphors for the relationship between movements of a 6DOF input device and changes in the position and orientation of virtual cameras. The metaphors (which they suggestively call "eyeball in hand", "scene in hand", and "flying vehicle control") were compared in three different environments and with respect to three different tasks. Their results showed complex interactions between these different study variables. Let us highlight the results of greatest significance to us. An enclosed maze seemed best explored and depicted as a movie using the flying vehicle metaphor for camera control, while the environment in hand metaphor made for confusing movies and poor exploration of this environment. A cube to be viewed externally yielded the opposite patterns of results with environment in hand manipulation (akin to the 'inspect' mode of many VR visualisers) being best for exploration and movie making.

Clearly, then, no one metaphor is best for all tasks. As we can imagine that participants in electronic arenas are likely to vary in terms of how they would wish to view objects (in relation to the kind of event they are participating in), we cannot opt for just one metaphor in our application domain. Even if we consider just one kind of participant (a virtual camera operator say), they are likely to have quite varied needs. Flying vehicle control would be important for many purposes but shots centred on objects of importance to the action may well benefit from inspection through an environment in hand interpretation of user input. For these reasons, the camera control interface developed for *OOTW* (described at length in Deliverable D7a.1) offered different methods of control including some redolent of Ware and Osborne's metaphors.

3.2.2.2. Simple Algorithmic View Control

Mackinlay, Card and Robertson (1990) examined real-time movement control in virtual environments and noted the prevalence, at that time, of interaction techniques supporting the rapid movement over long distances. Such high velocities tend to be hard to control when the viewpoint is proximal to an object leading to such familiar phenomena as overshooting. Mackinlay et al. propose a technique whereby target objects to be viewed are selected, whereupon the viewpoint moves with logarithmic slowing towards the object. Motion is initially rapid but slows as the object is approached. While this technique yields elegant controlled trajectories, it tends to require that the target object is already present in the field of view to facilitate ready selection. This certainly needs to be the case if objects are selected by some form of direct manipulation with a pointing device, as is the case with Mackinlay et al.'s implementation. Clearly, though, the technique could be extended to other implementations where, say, a destination is indirectly selected or computed and movement controlled by functions of which the logarithmic is but one.

Indeed, while we grant that logarithmically slowing one's approach to an object may give an elegant *first person perspective*, in an electronic arena, the choice of function may need to be influenced by considering how one's movement appears to other inhabitants (*second and third person perspectives*). For example, if I am engaged with a particular group of avatars interacting

with them, my departure may seem strangely and unaccountably swift if I suddenly move off to another destination of interest. My departure may need to be slowed too. In short, dynamical functions governing controlled movement are likely to vary as a function of the kind of participant in the electronic arena it is who has access to them. Thus, dramatic sweeping virtual camera movements may be obtained using Mackinlay et al.'s logarithmic slowing technique (they also discuss similar manipulations of the rate of change of orientation with respect to the target object). Indeed, the camera control interface developed for *OOTW* implemented exactly this in one of its modes. Other functions may need to be implemented for other participants so that their movements may be appropriately understood by others (see also the discussion of 'activity-oriented navigation' in Deliverable D6.3). (It is interesting to note that the cameras were not rendered in *OOTW*. That is, their movements could not be seen by other participants. We would imagine that the sudden acceleration of a camera as it departs along its logarithmically slowed trajectory might have confused inhabitants and performers if the camera had been rendered. This suggests that it is not merely what kind of participant you are which influences your repertoire of movement control functions, it is also how you should or need to be seen by others.)

3.2.2.3. More Sophisticated Algorithmic View Control: Procedures and Constraints

Drucker and his colleagues at the MIT Media Lab have developed a number of systems for the control of virtual camera movements. Drucker, Gallyean and Zeltzer (1992) describe CINEMA—a system which enables the definition of procedures specifying camera movements. Drucker et al. argue that direct manipulation user interaction techniques are not always appropriate when actions are likely to be routinely 'chained' and repeated or when a high degree of accuracy is required as might be the case for smooth camera movements. Accordingly, CINEMA allows the textual specification of camera movements in an extensible language with various levels of control from primitive movements through familiar camera types, such as dolly and crane, to more complex procedures based on relative position, direction of glance of actors and distances.

Drucker et al. show a number of small scale applications (e.g. a rendition of Hitchcock's famous shots from *Vertigo* which simultaneously covary camera position and field of view). However, no extensive evaluation in use of CINEMA is presented. Indeed, Drucker et al.'s paper closes with a clear recognition of the limitations of a pure procedural approach. They note that it is difficult to combine procedures or have multiple procedures simultaneously operating. This is a fundamental problem as quite simple applications might require multiple procedures to adequately specify an automatic camera movement, e.g. imagine a tracking shot constrained to avoid collisions with objects. For these reasons, the pure procedural approach seems limited to quite simple chains of movement and, accordingly, only a simple 'just in time' camera control mechanism along these lines was provided in *OOTW* (for details see Deliverable D7a.1). More complex interrelations of movement procedures may be possible in carefully staged and scripted applications but are likely to be unwieldy in real-time settings.

For their part, Drucker and Zeltzer (1994) go on to develop a system based on a different computational paradigm (constraint satisfaction) as an extension of their earlier approach. They introduce a concept of 'camera modules' which maintain local ongoing positional information and define how the camera is to translate user input (either directly or as defining camera constraints) while satisfying constraints which apply while the module is active. The notion that users need not directly manipulate camera positions but can set constraints is interesting but Drucker and

Zeltzer do not present any extensive examples of how this form of user-interaction might work in practice. Instead, they concentrate on formally specifying a camera module known as 'path' which computes a path through a 3D environment subject to constraints that the camera keeps a fixed height above the floor while avoiding collisions with objects.

Drucker and Zeltzer work through a museum application where paths are computed within and between rooms. An interesting algorithm is employed to avoid collisions. A 'penalty distance function' is propagated by 'wavefront expansion' from each object. This creates a 'map' of the repulsive potential of the environment which is combined, at run time, with attractive potentials towards the destination (e.g. the exit of a room) to produce a navigation function. The camera then moves in relation to this combined map. Computing the wavefront expansion, especially if there are many objects in the room to be avoided, is very expensive. Accordingly, Drucker and Zeltzer use a simplified 2D projection of the environment and precompute the 'repulsion map'. This, then, is not a pure real-time technique. While similar force-related navigational techniques have been developed which run in real-time (e.g. Hubbold and Xiao, 1998; Turner, Balaguer, Gobbetti and Thalmann, 1991), further consideration is required for environments where there are many *moving* objects which might (worse still) enter and exit the environment *during run time* at *unpredictable* moments.

In short, Drucker and Zeltzer's work, we imagine, will find itself most suited to non-real-time animation of fixed environments (perhaps the choice of a museum application is no accident!). Short of 'establishing' or 'geography' shots, it is hard to see how depictions of social interaction in an inhabited environment will be aided by wavefront propagation from static objects. Indeed, cameras might be hindered by following such a repulsion map as it is likely that the inhabitants will be avoiding those same objects too. This could increase the chance of the participant-camera collisions and occlusions that have been found to be disruptive in earlier inhabited television experiments and which the technologies reported in Deliverable D7a.1 were specifically designed to avoid. While the path module is ill-suited to animating cameras to capture social interaction, the concept of a 'mapped landscape' specifying forces which control cameras remains a fascinating one. Later in this chapter we shall present some ideas for camera control and deployment which, while not directly inspired by Drucker and Zeltzer's work, nevertheless have some affinity with it. (In fairness, of course, it was never Drucker and Zeltzer's intention to use the path module, or any other they write about, to capture social interaction in an electronic arena. However, this is precisely our point: while there is a rich literature with many interesting computational ideas for capturing uninhabited virtual environments on camera, there is little available, aside from our own work, devoted to electronic arenas which are essentially social. We make more of this absence from the literature shortly.)

3.2.2.4. Automating Viewpoint Placement

Phillips, Bradler and Granieri (1992) present Jack, a system to support the direct manipulation of objects in three dimensions by automating the placement of the viewpoint during the manipulation process. This helps avoid situations where it is hard for the user to see how rotations and translations should be performed because the geometry of the object projected at the user's viewpoint is unclear or ambiguous. While Phillips et al.'s techniques can be used interactively in real-time, it is clear that their use-scenario is one where users are engaged in exploratory manipulations of objects. Supporting this kind of activity is, of course, important and maybe useful for participants to an event in an electronic arena which involves 3D object

manipulation. However, it is unclear how essential such techniques will be to camera control and deployment in electronic arenas and other production/direction activities. The direct manipulation of the subjects of shots is not something we would imagine to be typical of such activities, or at least not something which would require dedicated support.

3.2.2.5. Through-the-Lens-Camera-Control

Gleicher and Witkin (1992) offer a paradigm for camera control which differs from others we have reviewed so far. It especially contrasts with the work of Ware and Osborne (1990) who are concerned to uncover the implicit metaphorical 'instrument' (eye, hand, vehicle) that the camera can be thought to be. Instead, Gleicher and Witkin are concerned with constraining or fixing features of the image as a way of controlling the camera: 'through-the-lens camera control'. For example, a pair of moving objects could be required to always be in shot at particular view coordinates. As the objects move so the camera moves all the time keeping the onscreen constraints satisfied. Gleicher and Witkin present an approach which circumvents a number of severe computational problems which would trouble many attempts to directly solve the equations reflecting the geometry of such situations. However, it is clear that their most prominent scenarios are non-real-time animations of uninhabited environments where the execution of object manipulation tasks is being visualised. To our knowledge, there are not yet real-time applications of through-the-lens control to inhabited virtual environments. This remains an interesting possibility as a number of recognisable shots from cinematic, TV and animation traditions could be supported this way (e.g. an elliptic tracking shot which keeps an interacting pair of actors in fixed relative locations on screen while moving the camera to show their surrounding environment, thereby economically depicting interaction and environmental context in the same shot).

3.2.3. Conclusions

Let us draw some conclusions from thus review of the most important contributions to the literature concerning camera deployment and control. We highlight six points:

- real-time operation
- mass social participation
- understanding rules of practice
- hybrid interaction methods in a working division of labour
- scripted and improvised action
- geometry, physical movement and capturing action.

We now discuss these in turn.

3.2.3.1. Real-time Operation

First, although real-time interactive VR systems are increasingly commonplace, much of the work we have examined has been developed with animation or other non-real-time applications in mind. Sometimes (as in Drucker et al.'s work) where real-time applications are targeted, considerable off-line computation is still required. We regard electronic arenas as typically involving events which essentially accent real-time interaction. This is the case with the most demanding of the inhabited TV and artistic applications we have examined in eRENA. Even if events in electronic arenas do have non-real-time components, it is worth prioritising real-time

applications for methodological reasons, the argument being that one can commonly 'pull back' from real-time techniques to asynchronous variants but techniques developed specifically for non-real-time application do not always generalise to real-time interactive settings.

3.2.3.2. Mass Social Participation

Second, while a number of application domains have been studied, few are adequately close to the applications we imagine an electronic arena will comprise. The demonstrator work in the eRENA project both on inhabited TV and mixed reality performances critically emphasises the real-time participation by a number (perhaps a very large number) of individuals. In contrast, most applications we have reviewed are single user and have no need to render participants at all, still less make active, mobile participants the subject of shots. The exceptions to this are very small scale (two or three co-interactors at any one time).

3.2.3.3. Understanding Rules of Practice

Third, we are unconvinced by the practical viability of some of the techniques in the light of what we know about the real world contingencies of professional work in inhabited TV or of implementing interactive media artworks. Ethnographic work in the project (see Deliverables D6.2, D7a.1, D7b.1 and Chapter 1 of this Deliverable) would gainsay the view that TV directors and producers, still less media artists, obey 'rules of cinematography' in composing sequences of shots or juxtaposing multiple visual projections. To be sure, practices of (say) continuity editing are well known and often oriented towards but this is not to say that they are slavishly followed. An approach to event design and management for electronic arenas which would assume that the rules of continuity editing have the same status as 'if... then...' computational procedures would be profoundly misguided. We do not see, therefore, that it is essential to build systems around such formal reductions of cinematic (or other) practice no matter how computationally tempting this may be.

3.2.3.4. Hybrid Interaction Methods in a Working Division of Labour

Fourth (and relatedly), our ethnographic research has persistently shown how interactive technologies for electronic arenas need to be developed to fit a 'working division of labour' between differently skilled participants to a production, between (say) camera operators and a director, or between a sound technician and a video tracking technician. We believe that technologies that offer varied combinations of manual and automatic (or 'delegated') control are most suitably flexible for complex co-operative work settings. This may often undercut the motivation for seeking fully automated solutions. In Deliverable D7a.1, we argued that technologies which had only partial degrees of automation (like the camera control and event management interface) allowed the director to experiment with different styles of editing and varied pacing across different shows. For emergent, experimental applications, this is surely correct. This is not to say that autonomous cameras (and other such automatic processes) have no role. On the contrary, we can see potential for automatically computed sources being available, from time to time, for a director to cut to if this yields material she judges as relevant and interesting. It is hybrids of this sort that we are interested in developing, hybrids where automatic and manual control coexist, and where humans can variably interact with, intervene upon or delegate control to autonomous processes.

3.2.3.5. Scripted and Improvised Action

Fifth, several of the applications we have examined assume some script exists which can guide, say, the activation of relevant camera modules. In electronic arenas, this may be so (cf. *OOTW* and how the event management and camera control software interact, Deliverable D7a.1) but it need not be so (*OOTW* and the artistic performance events in eRENA, e.g. *Murmuring Fields* and the performances described in Chapter 1 of this deliverable also have a strong component of unscripted improvisation). In many respects, the greatest design challenge for developing camera deployment and control technologies is to design for such improvised situations. If a system can be shown to be feasible in situations with a high degree of real-time unpredictability, then it is reasonable to imagine that they might also be workable when a script exists. In addition, when a 'script' does exist for action in an electronic arena, we do not wish to be confined to the computationally simple instantiations of the idea we have encountered in the literature. For example, we wish to allow for means of structuring an event which are more flexible than finite state machines (a topic we return to in Chapter 7).

3.2.3.6. Geometry, Physical Movement and Capturing Action

Sixth, a great deal of the work we have studied essentially involves the solution of the (sometime complex) geometrical problems which arise when a 3D environment is projected onto a 2D display. Optimising camera shots is, for many of the authors, fundamentally a geometrical affair or translatable into a geometrical problem. The virtual physical movement of a camera is also often conceived in terms of a trajectory along a path specified as a solution to a geometrical problem or in terms of a virtual physical potential. We cannot expect camera deployment and control conceived of *solely* in this way to fully address the requirements of social spaces such as electronic arenas. Consider again the direction of *OOTW*. While the director was naturally most concerned with the composition of shots, with appropriate arrangements of objects and avatars on screen, a more fundamental issue was the depiction *of action.* It was action (not objects in geometrically constrained arrangements) which was her concern. How best to follow it, capture it, display it to an audience. How to avoid missing it. How to maintain a set of options from the camera operators so that she would not find herself with nothing to cut to. These are not *directly* geometrical or virtual physical problems. For these reasons, in what follows in this deliverable, we devote ourselves to exploring techniques that are directly concerned with supporting what we call *activity-oriented camera deployment and control*. As we shall see, it is a concern to support production personnel in capturing action in potentially large scale, mass participation electronic arenas which guides the new work we present in this deliverable. If our review of existing literature on camera deployment and control in virtual environments is accurate, we believe that this is an innovative approach to the fundamental research problem of how views in a virtual environment can be configured and controlled.

3.3. Activity-Oriented Camera Deployment and Control

We seek to support event management in electronic arenas by facilitating camera deployment and control. We wish to do this through elaborating techniques which take into account the ongoing activity in the electronic arena, making this available (i) as a resource to guide personnel in their camera deployment decisions and (ii) to inform autonomous camera movement algorithms designed to seek out 'hot spots' of activity. Exactly how we do this will be discussed later in this chapter. Both uses of activity information require that data sources be found that can

serve as adequate heuristics for activity. The next subsections discuss what these data sources might be.

3.3.1. Activity Heuristics

We suggest that activity in an electronic arena might be made available in two basic ways.

- **Activity indicators**. By this we refer to traces of participant-activity which are available to whatever system it is that is maintaining the electronic arena. In a shared virtual environment, for example, it would be possible, in principle, to formulate some measures of communicative activity through, e.g. carrying out appropriate computations over keystrokes (for text communication) or audio-bandwidth usage (for audio communication). Equally, in virtual environments where objects are manipulated, some index of activity could in principle be computed on the basis of the prevalence of these interactions. Finally, for embodied activity in a mixed reality electronic arena, it would be in principle possible to use, for example, video analysis techniques such as mTrack or Wobblespace (see Deliverable D7a.1 and *passim* in this deliverable) to appraise activity and its locus. Note in all of the above, we say 'in principle' and refer to some 'appropriate computations' or measures being possible. Naturally, what measures should be used will turn out to be a critical issue and one which we must address to move from 'in principle' promises to practically viable techniques. Our own work has yielded interesting results using indices as simple as keystroke rate but, in particular practical applications, we imagine that finding the right indices and how to combine them will require much attention. (For further discussion on these points, see Deliverable D6.3.)
- **Awareness-based activity inferences.** A second way to heuristically determine where the action is in an electronic arena is to infer the patterns of collective awareness that exist within the participant-population and use this to infer, in turn, where action of interest is being or is likely to be realised. Let us explain this in more depth through an example. Imagine an electronic arena where performers are acting out an event in the style of promenade theatre, moving through a virtual environment as they perform. These performers and their actions will be the subject of attentiveness from the audience as the audience maintain an awareness of what the performers are doing. This attentive awareness is likely to be revealed by their positions and orientations around the performers, the directions of the gaze, and the correlated movements they undertake as they follow the performers. Naturally, in an example like this, knowing where the performers are at any moment may be resource enough to facilitate camera deployment but, in electronic arenas where interesting action might occur at any place and at any time, being able to make inferences about where this might be on the basis of the patterns of attentiveness and awareness among participants could be a viable approach. A number of virtual reality systems which are of potential use in electronic arenas actually implement an awareness model of some sort which could enable awareness information to be captured much like the activity indicators we have just discussed (e.g. the MASSIVE system used in several of the inhabited TV experiments, Greenhalgh and Benford, 1995). When this is not the case, an awareness model could still be superimposed on participant position and orientation data to infer patterns of awareness (indeed, this is the approach in much of our own work). Either way, further discussion is required as to how exactly notions of awareness

could be specified with enough formality to enable heuristic indicators of activity to be extracted.

3.3.2. The 'Spatial Model' of Awareness

The so-called 'Spatial Model' of awareness—largely developed in the ESPRIT project COMIC (1992-1995)—is one of the most ambitious attempts to provide multi-user cooperative systems with a notion of awareness which can shape information display to participants as well as their activities with information and interactions with each other (see Benford et al., 1994). As our own work builds upon this approach, we shall describe it in some depth. The Spatial Model supposes that objects (which might represent people, information or other computer artifacts) can be regarded as situated and manipulable in some space. The notion of space is very generally conceived only subject to the constraint that well-defined metrics for measuring position and orientation across a set of dimensions can be found. In principle, any application where objects can be regarded as distributed along dimensions such that their position and orientation can be measurably determined is amenable to analysis in terms of the Spatial Model though, naturally, virtual reality applications give a ready understanding of space in terms of 3D spatial geometries.

The interaction between objects in space is mediated through the relationships obtaining between up to three subspaces: aura, focus and nimbus. It is assumed that an object will carry with it an aura which, when it sufficiently intersects with the aura of another object, will make it possible for interaction between the objects to take place. On this view, an aura intersection is the pre-condition of further interaction. In many applications (our own included, see below), this helps with the management of scale as further awareness analysis need not always be performed. For objects whose aurae intersect, further computations are carried out to determine the awareness levels the objects have of each other. The subspaces of focus and nimbus are intended as representing the spatial extent of an object's 'attention' and its 'presence' respectively. Thus, "if you are an object in space, a simple formulation might be: the more an object is within your focus, the more aware you are of it; the more an object is within your nimbus, the more aware it is of you," and accordingly, "given that interaction has first been enabled through aura collision: The level of awareness that an object A has of object B in medium M is some function of A's focus in M and B's nimbus in M" (Benford et al., 1994).

It is important to note that in the above definition, awareness-levels are defined per medium. Thus, the 'shape' and 'size' of each of the aura, focus and nimbus subspaces can be different, for example, in the visual (graphical) than in the audio-medium. In this way, I may be aware of the sounds made by another object but without being able to see it. Benford et al. (1994, 1996) go on to show how simple instantiations of this model can have a high degree of expressive power, for example enabling one to distinguish between different intuitively familiar 'modes of mutual awareness' on the basis of A's awareness of B and B's awareness of A. However, perhaps the most important point emphasised in this work is the insistence that awareness is a joint-product of how I direct my attention to you (focus) and how you project your presence or activity to me (nimbus). The Spatial Model has influenced the fundamental architecture of a number of cooperative systems and has been extended in a number of ways by recent authors. Benford, Greenhalgh and Lloyd (1997) have introduced a concept of 'third party objects' which 'intervene' between objects and transform the nature and level of the awareness that objects might have—a feature implemented in MASSIVE and used to support a number of the details of interaction in *OOTW*. Rodden (1996) has reinterpreted the Spatial Model in terms of spaces that can be

represented as graphs of interconnected objects. Sandor, Bogdan and Bowers (1997) go yet further and generalise the Spatial Model concepts to apply to any semantic network of objects and their relations, and conceive of the aura, focus and nimbus subspaces not as bounded volumes in geometrical space but as the outcome of 'percolation processes' through networks.

3.3.3. Mapping Activity and Awareness

In an unpublished paper, Sandor and Jää-Aro propose 'activity maps' as representations of virtual environments based on activity heuristics such as those we have discussed (indicators like text/speech input measures, measures of avatar displacement, object manipulation and so forth). An activity map will be some analogic representation of the virtual spaces enabling a user to (for a visually rendered map) see a depiction of activity levels across a virtual terrain. Sandor and Jää-Aro propose that activity maps are computed by summing activity measures at each locus on the map. The number of loci differentiated in the map is the resolution of the map. A low resolution map (for example) might just show activity levels in North-East, North-West, South-East and South-West quadrants. As we have discussed, activity maps might also be inferred from the *operation* of an awareness model such as the Spatial Model (if a system implements it) or from its *application* (if a system doesn't). Separate awareness related maps could be computed for focus, nimbus or combined awareness measures based on joint focus/nimbus functions. A focus map would show the 'hot-spots' where, in general, the population of participants are directing their collective attention. The loci of promenading performers (see our example above) one can imagine would show high collectively summed focus levels. Conversely, a nimbus map would highlight the loci where participants are projecting their presence. Finally, a combined focus/nimbus awareness map could show the summed combinations of focus and nimbus at each locus to give a general impression of the distribution of collective awareness around the electronic arena.

Maps computed in this way would give an overview of activity and awareness in an electronic arena. Our proposal is that it is such maps which can potentially serve as a resource to production personnel in guiding directorial work—in particular camera deployment. However, it is very important that the design of a map for real-time human visual inspection is carefully considered. Our studies of production work in connection with *OOTW* would suggest that visual engagement with such a representation of activity would have to be very efficient with personnel being to pick up at-a-glance the relative distribution of activity in the environment. As such it would be important that map displays are not overly embellished with distracting visual detail and enable very swift interactive gestures in relation to them. Accordingly, we do not think that it would be appropriate to present personnel with anything other than 2D non-navigable displays. In the real-time of electronic arena direction, there simply is not time to navigate around a higher dimensional display. Indeed, the whole point is to give efficient overviews. For these and other practical reasons (cf. Bowers and Martin, 1999), we have designed displays for activity maps based around a very simple set of 2D shapes to represent participant locations. These shapes are colour highlighted to indicate the awareness levels at those loci. Our exact design will be presented under 'Implementation' below (Section 3.5).

3.4. Algorithms

3.4.1. Algorithmically Deploying Cameras: Identifying Groups

It is possible that readily interpretable activity maps would suffice to enable a director of an event in an electronic arena to, for example, give verbal instructions to a virtual camera operator ("head South-West and find out what's going on") or make simple manual deployments (e.g. to an approximate location in the South-West). However, we wanted to explore some design possibilities which might enable the algorithmic calculation of more optimal initial deployments which could, in turn, be manually refined by a camera operator. Our camera deployment algorithms are all concerned with finding a location and orientation for a camera in relation to a *group* of participant in an electronic arena. Given an identification of a set of group members, our algorithms return coordinates and a view vector for a camera so that a shot of the group can be optimally framed subject to certain constraints. We shall shortly present a formalisation of our algorithms.

We imagine that groups could be identified in a number of ways:

- **Explicit pre-definition**. Such a group is a set of participants who are defined at the outset to be group members (e.g. a football team or Aliens and Robots as in *OOTW*).
- **Explicit ad hoc definition**. Here, group membership could be defined on an ad hoc basis as was seen appropriate at the moment (e.g. a member of the production crew could, perhaps on inspection of an activity map, identify a group and explicitly select them).
- **Algorithmically mediated group identification.** One can imagine algorithms operating to identify groups on the basis of *geometrical information* concerning, e.g., the proximity of members, their common alignment, their mutual orientation towards a common centre, their common motion, and so forth. Alternatively, algorithms might identify groups on the basis of *activity or awareness information*, e.g. a set of participants who are all aware of the same object(s) could be taken to be a group. In Chapter 5, we discuss a way of using awareness information in combination with user input. The user indicates a location in the electronic arena and all the participants with an above threshold awareness level of this location are selected as a group.

In the implementation we describe in this chapter, the user can select a group in an explicit ad hoc fashion by the conventional means of drawing a selection region on the on-screen activity map display. Model' of awareness—largely developed in the ESPRIT project COMIC (1992-1995)—is one of the most ambitious attempts to provide multi-user co-operative systems

3.4.2. Algorithmically Deploying Cameras: Heuristics for Initial Shots

While we imagine camera angles ultimately have to be refined by a human camera operator, we suggest that they may be aided by software heuristics giving rough starting positions that then can be adjusted. We have defined three algorithms that seem to us to give reasonable initial camera deployments.

3.4.2.1. Centre of Gravity

This method adapts the technique used in the *OOTW* camera interface described in Deliverable D7a.1, Chapter 3, and in some respects generalises it (as the algorithm works with groups whose membership is determined ad hoc.

- Select a group of actors $G = \{g_0,\ldots,g_n\}$, according to the principles presented earlier.
- Define the centre of gravity *cog*(*G*) of the actors in *G*. The camera will be placed pointing towards the centroid.
- Determine the diameter *D* of the hull of *G*.[3] The camera should now be placed at such a distance *d* that the field of view *fov* will contain this diameter, $d = D/\left(2\tan\frac{fov}{2}\right)$. Note that since we are using a perspective projection for the camera, there might be objects outside the field of view anyway, the risk being larger for larger fields of view. But, as we noted above, this method will only give a rough position and in the projection display the field of view for the cameras is given and any adjustments are then easily made. A 'fudge factor' can be taken into account to automatically preadjust the distance, in our experience 1.2 seems to work well.
- The camera is now placed on a circle centred on the centre of gravity of the group, *cog*(*G*). The angle ϕ to be chosen should be the one that maximises the amount of 'faces' visible in the image—assuming that we want to capture actors communicating with each other, otherwise we should of course minimise the amount of face. So, to get face-on shots, we seek the maximum value of $\sum_{i=1}^{n}\sin(view(g_i)+dir(camera,g_i))$ where $view(g_i)$ is the direction in which actor g_i is looking and $dir(camera,g_i) = \arctan\frac{pos_z(camera)-pos_z(g_i)}{pos_x(camera)-pos_x(g_i)}$ is the view angle from the camera to actor g_i. Since we don't need a very accurate value, as we will manually adjust the position anyway, it is sufficient to just sample the function at, say, 0.2 radian increments and pick the angle giving the maximum value, which should be close enough to the actual angle.
- For from-behind shots we instead seek the minimum of this function. We can also get profile shots by using the cosine instead of the sine. If we wish to *favour* a particular actor *A* (i.e. make *A* the main subject of the shot), the term for *A* can be weighted higher.

3.4.2.2. Centre of Viewpoint

An alternative algorithm, which requires somewhat less computation, is:

- Determine *cog*(*G*) and *D* as before.
- Use the sum of the view vectors $O = \sum_{i=1}^{n} view(g_i)$ for camera positioning, such that if |O| is larger than some threshold (our implementation uses $|O| \geq 1$) the camera is placed at the distance *d* along *O*, otherwise the camera is placed at the same distance along the *up* vector.

The effect is that if the actors are all looking at the same object(s), the camera will be placed for a frontal/behind shot, but if the actors are turned towards to each other, we choose an overview shot.

3.4.2.3. Bisecting View Directions

If we search for a pair of actors communicating with each other in order to get a shot of them, we can use the following algorithm:

- Find a pair of actors g_i and g_j such that $\forall g_k, g_l : view(g_i)\ view(g_j) \leq view(g_k)\ view(g_l)$, ie a pair of actors that are facing each other 'more' than any other pair of actors.
- Find the intersection of the two view vectors $p = (pos_x(g_i) + view_x(g_i)\ t,\ 0,\ pos_z(g_i) + view_z(g_i)\ t)$, where $t = \frac{view_z(g_j)\left(pos_x(g_i) - pos_x(g_j)\right) + view_z(g_i)\left(pos_y(g_i) - pos_z(g_j)\right)}{view_z(g_i)\,view_x(g_j) - view_x(g_i)\,view_z(g_j)}$
- Place the camera along the vector bisecting the angle between the two view vectors for a 'with' shot, and along the negative vector for a 'towards' shot.
- Since the selected pair may be on the edge of the group, we can not use the diameter D in order to catch the entire group, as defined earlier. Instead we directly determine the maximum distance of any group member from the two we have selected, $D = \max_{\substack{g_k \in \{g_i, g_j\} \\ g_l \in G}} |pos(g_k) - pos(g_l)|$

The relative merits of these three algorithms, at least as we can currently appraise them, is discussed towards the end of this chapter.

3.4.3. Activity and Awareness Maps as Giving Dynamic Potentials to Autonomous Cameras

So far we have discussed how certain kinds of initial camera deployment can be algorithmically determined given a selection of a group of participants to make the subject of the shot. We imagine that personnel deploying cameras would make such deployments in the light of inspecting an activity map or some analogous overview, perhaps by explicitly selecting the group themselves. We have also explored in preliminary form a camera type whose behaviour is more closely related to the nature of a computed activity map. We have experimented with an actively *activity-seeking camera*, which we refer to as 'puppycam' as its behaviour is in many ways redolent of a young puppy always seeking out matters of momentary local interest. Conceptually, the puppycam will follow the gradient of an activity/awareness function as mapped on the basis of heuristically derived activity indicators, awareness measures or whatever. In our experiments so far we have applied an awareness model onto participant orientation and position data to yield an awareness map. Our puppycam will always move in the direction of increasing awareness. The intention is that it will find the area with the highest activity, as given by the heuristic that high awareness levels also correspond to high activity levels.

The actual implementation at every timestep samples the awareness function at twelve points on a unit circle surrounding the camera as well as at the camera position itself and then moves the camera to the position which has the highest awareness value, facing in the direction of increasing awareness.

There are certain problems with the puppycam in this implementation. The most important is that the spot of highest awareness often is quite jittery, as small movements of the surrounding

participants get translated into awareness values. If the puppycam is undamped it will then often start oscillating around the awareness maximum, swinging its view direction back and forth every timestep. A damping function, such as the one described in Turner, Balaguer, Gobbetti and Thalmann (1991), is necessary to get any kind of image coherence.

Another problem is that if several puppycams are in use, they may all be attracted to the same spot. Note that this is not necessary outcome, since they only seek out local maxima of the awareness function but it is a likely outcome for awareness 'terrains' which are relatively 'flat'. Not only does this seem a waste of camera resources, we documented in Deliverable D7a.1 who the director of *OOTW* described the phenomenon of all cameras heading to the same shot as her 'worst nightmare'. However, we can easily define the puppycams to themselves have negative nimbus values, thus in essence 'scaring each other away' and maximising variety in the shots they provide.

A further enhancement of puppycams, making them even more similar to young dogs, is a 'boredom counter'. When a camera has been at the same local maximum for a certain length of time, it will become increasingly more likely to seek out a new maximum. This is most easily implemented by incrementing a counter for each timestep the puppy has not moved and subtracting that counter from the current awareness value, thus making the current point less attractive over time.

We can extend this discussion of puppycam to identify a general class of dynamic algorithmically driven cameras which sample the local awareness/activity gradient around them. In general, we imagine autonomous cameras whose behaviour is governed by three 'forces' or 'tendencies':

- how they translate their sampling of the local awareness/activity gradient into a potential displacement
- the extent to which they are attracted or repelled by other cameras
- the extent to which they persevere at their current location.

We imagine that an interesting class of dynamical behaviours for autonomous cameras could be defined using just these three parameterisable tendencies. As such our discussion bears comparison to the 'forces' often postulated in the simulation of 'flocks' and other collective entities (see the simulation work in Deliverables D5.3 and D6.3). However, to our knowledge, the notion that a set of multiple cameras in a dynamic multi-participant virtual environment could be conceived of as 'flockmates' is original here.

3.5. Implementation

We have developed a prototype implementation of the principles we have described, called SVEA (Sonification and Visualisation for Electronic Arenas). The sonification component of SVEA is described in the next chapter. Here we concentrate on how SVEA implements the Spatial Model, how it computes and displays awareness maps, the support for simple mouse-based interaction we have included, and various other usability features.

3.5.1. Interpreting the Spatial Model in SVEA

One interpretation of the Spatial Model is to define the focus and nimbus as actual volumes associated with their objects, and the awareness as the (normalised) intersection of these volumes. In other words object *a*'s awareness of object *b* is defined as

$$awareness(a,b) = \frac{\iiint_{focus(a) \cap nimbus(b)} 1 dxdydz}{\min\left(\iiint_{focus(a)} 1 dxdydz, \iiint_{nimbus(b)} 1 dxdydz\right)}$$

However, this is sufficiently computationally complex as not to be suitable for real-time applications. A better suggestion is therefore to define the focus and nimbus as functions that take on a value in the interval [0, 1] for any point in 3-space. In this case the corresponding awareness computation would become $awareness(a,b) = focus(a, pos(b)) \;\; nimbus(b, pos(a))$.

Note that this latter definition allows focus and nimbus to take on continuously varying values, whereas in the former a point is either within focus/nimbus or not. In general, the *pos* function can be intentionally left without a stricter definition, so that the formulation is useful both for continuous 3-space, as well as in discrete graph spaces. In the implementation we have worked with most, though, *pos* returns euclidian distances, and the focus and nimbus functions are simple inverse functions of distance with, at the extremes, (i) a cut-off at 50 units of distance and (ii) a distance of 0 units (i.e. two objects which are coincident) yielding focus and nimbus function values of 1.

In current implementations, we do not work with an aura concept but, in general, we would like to avoid performing awareness computations between all pairs of actors, as this scales badly. Thus, it will be useful to define the aura as a bounding volume for the focus and nimbus of an actor and only perform awareness computations for those actors whose aurae intersect. We can then use collision detection optimisation techniques (Fairchild 1994, Bandi 1995) to improve performance.

3.5.2. Computing and Displaying Activity Maps in SVEA

It is interesting to note that social virtual environments are almost invariably designed with a single or a small integer number of two-dimensional planes, even though avatars may be mobile in three dimensions. One may speculate that one reason for this is that many of these worlds are designed as simulacra of the real world, which much of the time is relatively flat (see Figure 3.1 for a sampling of social virtual environments). Similarly, Bowers and Martin (1999) present a number of arguments, from a social scientific point of view, for why relative flatness of environments and a non-equal treatment of the three dimensions in terms of the distribution of virtual objects (e.g. so that objects tend to cluster around planes) might be preferred in many cases. In a number of the artistic works developed in eRENA, we still find a predeliction for flatness. For example, a work like *Murmuring Fields* (see Deliverable D6.2) is more concerned with the juxtaposition and superimposition of multiple 2D depictions within a 3D environment than with filling out an environment with graphical material equally in all directions. Finally, we note that in inhabited TV applications, environments have tended to be designed so that *either* activity remains on the ground plane thereby simplifying camera work and reducing the number

of degrees of freedom required for participants' interaction devices (*OOTW*), *or* activity is restricted to multiple 2D 'levels' (*Heaven and Hell - Live*).

Figure 3.1: A set of relatively flat social virtual environments: Active Worlds, Blaxxun, and Out Of This World.

For all of these reasons, it is often a tolerable simplification to work with 2D activity maps which show the location and orientation of participants on a groundplane (perhaps selected from a set of levels, if the environment is so designed).

We place isosceles triangle shaped markers representing the participants of the electronic arena on a 2D projection of space with the colour of these markers displaying a measure of the activity they show. Areas of high activity will thus be conspicuous as large, brightly-coloured areas. While SVEA can use a variety of different kinds of data to give activity measures, we default to applying the awareness model just described. We give a colour to a participant P's marker which is in relation to the sum of awareness that all other participants have of P—the greater this figure, the brighter the colour.

The sharp apex of the triangle is used to point to the participant's location, with orientation being represented so that the triangle can be understood as an arrow pointing in the direction the participant is facing. This has the consequence that a distinctive 'flower' shape can sometimes be seen as a group of participants aggregate and face inwards.

The 2D display can be magnified up to 16 times by selection of a menu option to zoom in on a selected group of interest with the centre of that group being the centre of the zoom. Zooming rescales the relative separation of markers in screen distances but not the size of the triangles themselves. Zooming, therefore, can clarify the relative positions of participants which are so close to each other as to appear overlapping when in a 'wide angle' view.

3.5.3. Camera Deployment and Real-Time Interaction with Activity Maps in SVEA

In addition to the markers depicting participants, a set of cameras, representing possible viewpoints in the environment, is displayed. By default, the cameras will be activity-seeking puppycams, i.e., they will move towards areas of high activity. Cameras can be selected, with the intended semantics that the view from that camera is the transmission (TX) view. SVEA can be connected to a DIVE visualiser (see http://www.sics.se/dive/) that will enable a 3D visualisation of the electronic arena to be obtained from the selected camera.

Algorithmic camera deployment is actioned by dragging the mouse over the display to select a set of markers. As soon as the mouse is released a camera is deployed to the algorithmically computed optimal location for that group according to whichever algorithm is set as a preference. In this, algorithmically enhanced way, camera deployment can be efficiently actioned by a single interface gesture.

In principle, SVEA can take a real-time stream of data concerning participant position and orientation and visualise inferred awareness levels. In our experimentation to date with SVEA, we have worked with data logs from actual inhabited TV events (e.g. the *Heaven and Hell - Live* data) or with similarly formatted logs from the simulations conducted in Workpackage 5 (see Deliverable D5.3). Jason Morphett at BT Labs has kindly provided these data logs that we read into SVEA via an autonomous thread to simulate the real-time arrival of data at a socket. As an alternative way to control data log input, a 'time slider' is provided at the base of the SVEA display to move backwards and forwards through the data. Thus, we have equipped SVEA was basic tools to support the off-line browsing of activity in an electronic arena as well as real-time action on the basis of it.

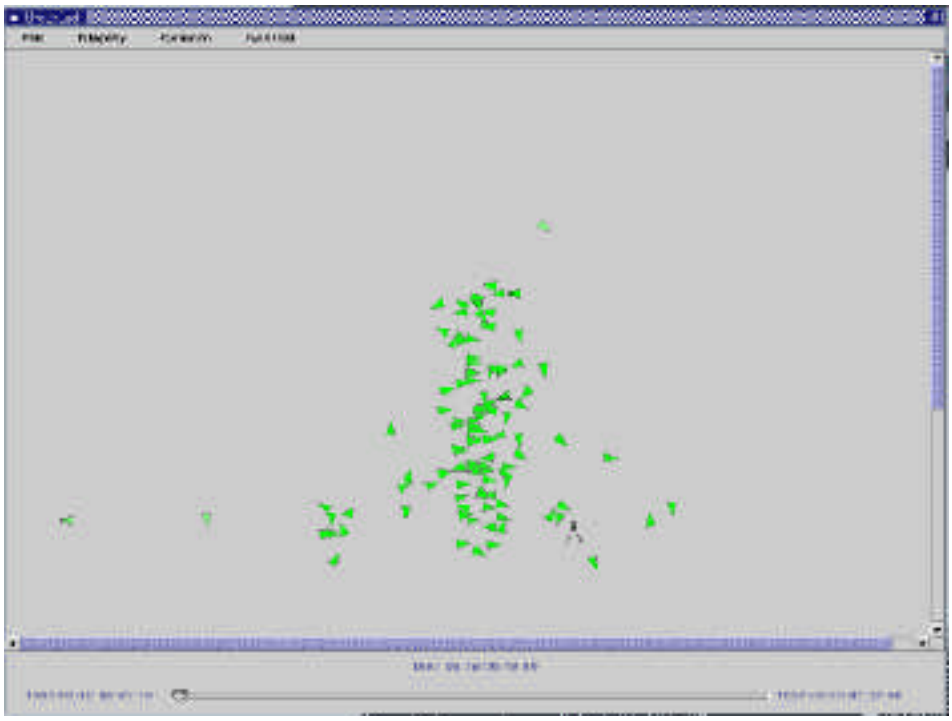

Figure 3.2: SVEA - a pre-recorded data file has been read in and is visualised.

SVEA has been implemented in Java 1.2 and Swing for maximal portability, and while this promise has not been fulfilled in all respects, we have run it with acceptable performance under

Solaris 5.6, Irix 6.5, Windows 95, 98 and NT. An important development of SVEA is discussed in Chapter 5, where we describe the alternative physical interfaces we have built to the core of the application. This enables group selection, camera deployment and selection, and display zooming through the manipulation of physical icons (phicons) on a table-top projection surface. Amongst other matters, this facilitates some of those aspects of the interaction with SVEA, which—in its GUI realisation described in this chapter—are supported by somewhat cumbersome menu selection operations (e.g. display zooming).

3.6. Future Work

At various junctures in this chapter we have already indicated some future development possibilities (e.g. including an aura concept into SVEA's awareness model to enable scaling up to large participant numbers and high data input rates). We anticipate development of SVEA to continue throughout Year 3 of eRENA with the key accents being: (i) assessing and promoting the usability of the application, (ii) enhancing its integration with other applications and techniques being developed in eRENA, and (iii) fixing a number of features we already know are problematic.

Currently, SVEA has not been formally evaluated for its usability in a work-like situation. That is, we do not yet know of its acceptability as a potential production tool or whether—in the exigencies of a real-time event—it actually will facilitate camera deployment and direction. To this end, we are planning a work-like evaluation of SVEA—particularly in its incarnation in the round table working environment described in Chapter 5 where more details of our evaluation intentions can be found.

There are a number of promising lines of integration with other developments in eRENA which our work is now well placed to engage in. For example, Deliverable D5.3 discusses some collective behavioural modelling techniques under development at BT Labs and EPFL which have as part of their motivation the aim of predicting behaviour in electronic arenas so as to, for example, allocate networking and system resources intelligently. The same simulation techniques could be used to provide predictions of future participant positions, orientations and activity so as to deploy cameras pre-emptively. That is, in a real-time event, it might be possible to jump off-line and push the visualisation into the (predicted) future a little, assign some cameras to the imminently likely hot-spots, and then return to the present. The slider at the bottom of our display, then, could actually be pushed into the future to acquire simulation data—even as real-time data is being received!

Our work on the notion of 'activity-oriented camera deployment and control' has enabled us to speculate about generalisations of our approach to individual and group navigational issues (see Deliverable D6.3)—a matter of cross-workpackage integration that we intend to enhance in Year 3. Chapter 5 of this deliverable, where we report the round table environment and our physical interfaces to SVEA, is also an example of cross-workpackage co-operation.

The work in Chapter 5 also gives a clue as to how our technologies can be integrated with the interest in mixed reality environments in eRENA. On this topic, it is important to emphasise that, although our work takes as its main focus the deployment of virtual cameras in a virtual environment, extensions of our techniques are certainly possible to mixed reality electronic arenas. We have already speculated that methods of video analysis could be used to gain activity measures that could be input to a SVEA-style visualisation concerning activity in a physical

environment. Selections from such a visualisation might deploy resources appropriate to a mixed reality setting. For example, in a large-scale physical environment with multiple projection screens and sound systems, a representation of participant activity might be interacted with to influence sound diffusion or decide the distribution of image to screen. Although our application is concerned with virtual camera deployment, the essential concept of using visualisations (and in the chapter that follows, sonifications) of participant activity as an aid to real-time decision making over resources in an electronic arena is a general one.

Let us finish with a listing of some important, but less ambitious, tasks for future work. It is necessary to improve the interoperation of SVEA with DIVE, the VR system we use to provide 3D visualisation. In particular, we need to capitalise on the improved new Java-DIVE interface provided by Jive. We also need to test the interoperation of SVEA with MASSIVE. The sonification reported in the next chapter needs to be more thoroughly integrated with SVEA so that the combination of visualisation and sonification can be more systematically investigated. Further work needs to be done to investigate the appropriate combination of activity indicators for effective use in SVEA.

Finally, we need to implement more flexible ways of taking 2D planar sections through 3D virtual space. Currently, we just ignore the 'up' co-ordinate and project onto the XZ plane. This is far too crude. Instead, we need to explore such devices as a spatial 'range filter' which would enable us to identify layers in 3D virtual space and act upon them. While we remain convinced of the value of 2D projections as activity maps for settings which involve 'grounded' or 'layered' activity, we need to explore techniques for the construction of non-projective 2D maps. These might be appropriate in just those cases where users of SVEA need the efficiency of 2D interaction but the real-time position, orientation and activity of participants truly is distributed in 3D in a more isotropic fashion than we have observed to date.

Chapter Four
Supporting Event Management by Sonifying Participant Activity

John Bowers, Sten-Olof Hellström and Kai-Mikael Jää-Aro
Royal Institute of Technology (KTH), Stockholm, Sweden

4.1. Introduction

It was hypothesised that a sonic display of participant activity in a large-scale electronic arena would be a potentially useful resource for directors and producers of electronic events. This chapter reports on our initial explorations of sonification for electronic arenas, describing the development of sound models for representing participant activity and the initial experimental tests that these models have undergone. In this introduction, we give a brief overview of the research field of sonification and give details of our motivation for exploring sonification in eRENA. Later sections discuss our implementation, empirical studies and present possibilities for future work.

4.1.1 The Research Field of Sonification

Sonification is concerned with the use of sound to represent data much as data visualization is concerned with the analogous use of graphical displays. An appropriate representation in sound should enable the listener to understand relevant features of the data-set so as to pick up information about it. In short: “sonification is the use of nonspeech audio to convey information” (Kramer et al., 1997) or in a slightly longer formulation from the same authors: “sonification is the transformation of data relations into perceived relations in an acoustic signal for the purposes of facilitating communication or interpretation”.
Sonification has emerged as a research field rather recently. Kramer (1994) presented the first major published collection of various explorations in the field and a series of conferences (The International Conference on Auditory Display, ICAD) has provided a focus for the research community since 1992. Naturally, though, many applications of auditory display predate coining the term ‘sonification’, including: the Geiger counter, sonar, the auditory thermometer, and numerous medical and cockpit auditory displays (Kramer et al., 1997). There are many legends in the computer science community of researchers using sound to identify malfunctions in computers or to debug programs. One story concerns listening to a computer, which was running a supposed random number generator, using an AM radio. An audible beat pattern indicated that the numbers were not entirely random. The website http://www.santafe.edu/~icad/ is a useful resource for sonification, and other areas of auditory display research, and contains on-line versions of all the ICAD conference proceedings.
Researchers have examined sonic data displays in many application areas. Fitch and Kramer (1994) present a tool for the sonification of various features of the real-time condition of a medical patient. A simulation study with medical students of a version of a tool in which six different data dimensions were sonified suggested that emergency situations could be identified more reliably with the tool than with vision-only displays of the same data. Pereverzev et al. (1997) report on two physicists who were readily able to detect quantum-level phenomena in auditory displays of their experimental data that could not be detected in visual oscilloscope

traces. Applications for visually impaired users have been presented by, amongst many others, Stevens and Edwards (1997) and Lunney and Morrison (1997) while educational applications to facilitate the teaching of, amongst other topics, statistics (Flowers, Buhman and Turnage, 1996) have also been explored.

Sonification applications vary in the 'directness' of the link between data and its sonic representation. Since the early 1960s, researchers have explored the sonification of seismic data by, for example, replaying seismic recordings at audio rates, thereby overviewing a day's worth of data in a few minutes. Hayward (1994) reviews such research and presents a number of techniques for the presentation of seismic data. The term 'audiation' is sometimes used to refer to such 'direct' transformations of data into sound. Most applications, though, involve a less literal presentation of the data. Various techniques of sound synthesis have been explored to abstractly convey data relations with, typically, dimensions of variation in the data being mapped onto parameters in the sound model (e.g. Scaletti and Craig, 1990). This is sometimes referred to as using sound 'analogically' (Kramer, 1994).

Alternatively, sound can be used to, for example, represent features of the data-set in a 'symbolic' fashion (Kramer, 1994). Fitch and Kramer (1994) use a caricature of a breath sound to depict features related to a patient's breathing and Gaver (1994) has explored various techniques for designing 'auditory icons' which might, for example, symbolize the size of a filestore or a user's interaction with it.

Kramer et al. (1997) optimistically present sonification research as the necessary next step (after data visualization) to help us comprehend complex data-sets: "Although scientific visualization techniques may not yet be exhausted, some believe that we are approaching the limits of users' abilities to interpret and comprehend visual information. Audio's natural integrative properties are increasingly being proven suitable for presenting high-dimensional data without creating information overload for users. Furthermore, environments in which large numbers of changing variables and/or temporally complex information must be monitored simultaneously are well suited for auditory displays".

4.1.2. Sonifying Participant Activity in Electronic Arenas

The literature on sonification commonly documents a number of features of auditory displays which 'naturally' predispose their suitability to various applications (Kramer, 1994). For example, it is often argued that the human auditory system is particularly sensitive to changes in auditory display. Thus, tasks that require users to be *alerted to significant changes* are well suited to auditory applications. Human listening is essentially temporal so it is often argued that *time-varying data* is naturally suited to sonification. Auditory displays are *'eyes-free'* so that, in principle, complex visual tasks can be engaged with simultaneously and without interruption. It is commonly remarked that sound is *ambiently available* in a way that visual information is not. For example, we do not need to 'turn our ears' towards a sound to hear it in the same way that we have to turn to face a visual detail. The ears can lead and the eyes follow (cf. Kramer, 1994). Equally, as sound is ambiently available it is, headphones notwithstanding, naturally available to all in a shared environment and hence can serve as an informative resource for *cooperative work* (cf. Gaver, Smith and O'Shea, 1991).

While we do not want to dispute the existence of certain 'natural' features of sound or take issue with claims about the nature of the human perceptual system, the promise of an auditory display can only really be realized in particular applications designed for particular settings. Sonfiying time-varying data does not guarantee a successful tool or one more usable than a visual display. The mere ambient availability of sound does not guarantee that it can effectively support groups

of people working cooperatively. Everything hinges on the precise details of design and the practical activity designed for.

With this caveat in mind, though, we feel entitled on the basis of the literature on sonification and what we can anticipate about the production requirements of electronic arenas to hypothesize the applicability of sonification techniques to support the production of electronic events. Consider some of the findings and the emphasis of the ethnographic study of inhabited television reported in Deliverable D7a.1. In the real-time of direction of a live event, the director and other participants are commonly seen to be intensively engaged visually to some display or another. The director continually inspects her TX and camera monitors, only occasionally glancing away to her written running order or glancing down at the mixer desk in front of her. The camera operators are closely visually engaged with their camera interfaces, only occasionally glancing away to take in the views on the other operators' screens. Not surprisingly of course for a visually presented event (this is inhabited TV, not radio!), the ongoing status of visual events is monitored extensively by all. This suggests that parties to events with a production role might benefit from the promise of *eyes-free* interfaces. If they are to have further resources at their disposal to inform, say, shot selection or camera direction, then an auditory display which would not require disengagement from any visual display might be appropriate.

We have seen (again examine Deliverable D7a.1 on the phenomenon of 'vision following sound') that production and direction personnel are attuned to the *alerting-cueing role* that sound can have. When cutting dialogue or other forms of spoken exchange, a change of speaker can cue a cut to a different shot. As we shall see, we have it mind that production personnel might be cued by audible changes in a sonic display to cue camera deployments. That production personnel are attuned to listening for cues in this way gives us provisional assurance that an enhanced auditory display could be used.

Deliverable D7a.1 documents the *cooperative work* involved in producing and directing inhabited TV. Again we imagine that an ambiently presented sonic display might serve as a shared resource so that, for example, a change in display could cue a director to give instructions to a camera operator in a concise and mutually understood way (e.g. "search out what's caused that change", indexical references—"that change"—being possible because the auditory display is shared).

We wish to go further than just hinting at the in-principle applicability of auditory displays for some envisaged future electronic arena 'production suite' and focus on an exact application. This application is on the same general terrain as our visualization work described in Chapter 3. That is, we are fundamentally concerned with the representation of participant activity in electronic arenas so as to support the production and direction of electronic events. We intend to develop sonification tools that complement the visualization tools we have explored. In this way, features of participant activity which it is hard to visualize (or hard to depict with our current commitment to simple 2D visual displays) can nevertheless be made available. Participant activity in any electronic arena of interest is essentially *time-varying* and the depiction of activity (and changes in it) is precisely what the director of *OOTW* expressed a need for (so that, for example, cameras could be taken to where the action is). We hope therefore that a sonification tool can assist in addressing this practical need.

While we feel that participant activity in an electronic arena is an appropriate kind of data to sonify, we are cautious about the claims sometimes made in the sonification literature about the number of dimensions in the data which can be simultaneously represented in sound. For our application, and its setting (the potentially real-time direction and production of events in an electronic arena), we feel that a carefully limited number of dimensions is appropriate. If

listening to an auditory display is the only ongoing task, then one can imagine that many dimensions can be sonified using a complex sound model. However, we envisage our application fitting in with the multiple other tasks that producers and directors may have to do concurrently. Our auditory displays will not get the careful attention that two physicists might give their quantum data. Accordingly, we have restricted our application to just seven dimensions. Let us now unfold more of our design rationale in the sonification work we have done.

4.2. What Is To Be Sonified?

To prototype a sonification tool for representing activity in electronic arenas, we worked with the same *Heaven and Hell – Live* data-set used in our visualization work (though it must be emphasized that none of our applications are restricted to only this data-set). However, rather than represent the data directly (i.e. rather than 'audiate' the data-set), we explored whether sound could be used to represent simple summaries of the activity as captured in the data-set. Our visualization work was concerned with simple representations of the 'raw' data, *our sonification commonly presents the ongoing changing values of simple statistics*. For example, while our visualization presented the positions and orientations of each participant, we sonified how 'scattered' the distribution in space of the participants is at a given moment. Similarly, by inspecting the visualization over a brief time interval, one can gain an impression of how much movement around the electronic arena there is. This though is a matter of 'visual inference'—a matter of seeing patterns of change. In our sonification, by contrast, the overall amount of movement at a given time is calculated and sonified. In this way, *summary features that would have to be inferred from the visualization are calculated and represented in the sonification.* This is one strategy for making our sonification complement the visualisation. Another is that we try and compensate for weaknesses in our current simple visualization design. For example, to give a sense of the orientation of participants, the visualization employs small triangular representations. Though small, these have to be large enough for the orientation to seen. This can lead to occlusion problems. Especially when the population of participants is densely packed around a certain point, it can be hard to see just how many there are. Accordingly, we give a sonic representation to overall participant number. Our visualization strategy of ignoring the 'up' coordinate, though a viable simplification of the data and justifiable from a usability standpoint (see our brief discussion about the relative merits of 2D and higher dimension displays in Chapter 3), does have the consequence that up/down displacements do not effect the visualization. Equally, 'turning on the spot' is not so readily noticeable as a lateral displacement. These issues also guided the dimensions we chose for sonification. Our 'overall movement' statistic enables such changes to be heard if not seen. In this way, features which are not obvious given a certain repertoire of graphic resources for visualization can still be represented in sound; *absences or weaknesses in the visualization format can be compensated for*.

As soon as we make visualization into an interactive display, some relationships between data will inevitably be made more obscure at the very moment that others become clearer. A scrollable or zoomable display will be used to bring features of visual interest in the data into focus for closer examination. Inevitably, this has the downside that some data will be momentarily lost from the visualization and, possibly, phenomena of interest missed. Equally, zooming will mean that our sense of 'scale' and 'spacing' may not be constant from one view to the next. Unless there are very explicit interface devices used to indicate the current degree of zoom, users may make errors in their judgements of the relationships between data (for example, two participants may seem very far from each other just because an inappropriate zoom level has

been selected). We have made our auditory display of the participant-data non-interactive and deliberately so. As such, it can form a *sonic baseline against which changes in visual display can be understood* and *representation is still given to data that currently are not being visually displayed*. Accordingly, we hope that unseen developments of importance might still lead to audible phenomena in the sonification. In this way, *a sonification can compensate for absences brought about by the ongoing interaction with a visual display*.

4.3. Criteria for the Design of Sound Models

With a restricted number of dimensions to sonify (in our case seven), it is possible to design sound synthesis methods (sound models) which attempt to maximize the perceptual clarity of how changes in the data are represented. To help us do this, we followed the following principles that we believe to be of general interest and utility. The first two are, in many respects, the superordinate principles to which all others contribute.

- A sound model should be rich enough to support the representation of seven dimensions of variation in the data through seven associated control parameters.
- A sound model should be selected such that its parameters are clearly discriminable. When a parameter is varying it should be perceptually clear which it is (between parameter discriminability) and a range and scaling should be selected for it so that different values of the parameter are maximally different from one another (within parameter discriminability).
- Emergent percepts should not be misleading. It is a common phenomenon for complex varying sounds to yield simpler percepts than a physical description of them might lead us to believe, especially when the interaction between multiple parameters have emergent effects. Bregman (1990) and other authors have pointed to ‘auditory streaming’ phenomena whereby the perceptual system simplifies multiple independent auditory streams into a smaller set. A sound model should be designed so that such streaming and related effects, if they do occur, potentially signal phenomena of interest in the data.
- ‘Symbolic’ and ‘analogic’ sound synthesis are to be combined. Some of the dimensions to be sonified concern the communicative activity of persons as they interact with each other in the electronic arena. It seemed natural to use symbolic means to sonify this. Oscillators and filters with vocal characteristics (to enable a simple form of speech synthesis) were used in the sound model to convey these aspects of the data.
- The various components of the synthesized sound are designed to be heard clearly at low amplitude levels. This should enable the sonification to be deployed without it becoming excessively intrusive, annoying or disruptive of other auditory tasks that users may have to engage in.
- Only one data dimension should cause variation in the bass range of the sound model. It is easier for listeners to perceptually distinguish multiple auditory streams in higher frequency registers than in lower frequency ones (cf. Bregman, 1990).
- Only one data dimension should be represented as a continuous drone sound. More than one could cause interference (e.g. 'chords' which need not reflect meaningful relationships in the data). The other dimensions all yield streams containing sounds that are short in duration

relative to the silence between them. Again this minimizes potential confusion between sound streams.

- One data dimension should be represented by varying perceived spatial location in a multichannel sound system (our sonifications are two channel, i.e. stereo). Two dimensions can be represented, but only if the sound streams associated with the dimensions are highly discriminable in terms of their non-spatial characteristics.
- While we do vary spatial location in our sonification, it is important that the effects of the sonification are not designed to be specific to a single listening position. Recall that we intend our sonification to be heard by a group of people cooperatively working on the production and direction of events in an electronic arena. This militates against the use of headphones dedicated to the sonification or the optimisation of a stereo sound system for a single listening position and an immobile listener.
- Notable spectral changes (i.e. changes in frequency content or 'timbre') should be used for the sound stream associated with just one data dimension.
- Notable amplitude changes again should be used for just one sound stream.

As a final principle, it is important to bring out a consequence that our above requirements have for how the model should be implemented:

- Techniques of sound synthesis should be preferred over the use of sampled sound as synthesis enables greater control of a sound model at all levels of detail. The sound synthesis engine we use does not have a fixed synthesis architecture unlike most commercial synthesizers (see below). This enables us to configure individual oscillators, filters and so forth ad lib and experiment with different patterns of external control of the sound models we implement.

4.4. Implementation

A variety of sound models, consistent with the above principles, were implemented using the Clavia DMI Nord Modular sound synthesizer. This hardware synthesizer has DSP boards which are externally reprogrammable by means of an editor application which (up to version 2, August 1999) runs on an external PC which communicates editing instructions via MIDI system exclusive data. Though a hardware device, the Nord Modular's OS is software upgradable and an increasing variety of 'modules' are supported. These include a variety of audio-oscillators, filters, control-oscillators, audio and control modifiers, mixers, logic operators, external input and output modules and so forth. For experimental purposes, we have found a mixed hardware/software approach to sound synthesis (like that exemplified by the Nord Modular) preferable to either software only sound synthesis or reprogramming conventional sound cards in terms of flexibility and reliability.

We intend our sonifications to work in concert with our visualizations. However, interfacing our Java-implemented visualizations to enable MIDI control of the Nord Modular presents a major problem. Though there are secondary uses of MIDI for inter-application communication or for internal control of sound cards or samples contained in OS-extensions (e.g. the QuickTime Musical Architecture), MIDI is fundamentally designed as a protocol to enable hardware devices to communicate with one another. This hardware-relatedness makes it 'philosophically' at odds with many traditional modes of thinking regarding Java. At the time of writing (August 1999),

the JavaSound API does not adequately enable the control of sound synthesized on a machine external to the machine that is hosting the Java program. That is, the control of MIDI through an external port is incompletely supported. Many traditional uses of MIDI, then, are unsupported by official Java standards at the moment.
Accordingly, we had to employ unofficial Java classes for MIDI communication. A variety of implementations exist. The majority of our work has used the 'nosuchMIDI' classes that are specific to Windows machines (see http://www.nosuchmidi.com), though we have also examined the MIDIshare Java implementations which support an impressive variety of platforms (http://www.grame.fr/MidiShare/). With nosuchMIDI interfacing between Java and MIDI communication with the Nord Modular, we have experimented with three different sound models.

4.4.1. A Pulse and Voice Simulation

This uses pulsed sounds with varying tempos and spectral contents alongside synthesized vocal sounds to represent data dimensions. It is the main model we have explored and it is the one we have formally tested for perceptual discriminability in the experimental studies described later in this chapter. In outline, it works as follows:

- The greater the number of participants in the electronic arena, the higher the overall amplitude of the sound.
- The more participants there are that are communicating with each other, the faster the synthesized voices 'talk' and the more synthesized voices there are to be heard. If one participant is currently communicating, we hear one synthesized voice. This speeds up to represent the simultaneous communication of two, three or four participants. Five participants are represented by a second synthesized voice entering. The two voice sound is increased in tempo to depict values between five and fourteen. Fifteen or more people are represented with three synthesized voices, again increasing in tempo with increasing number. This combined use of numbers of synthesized voices and their tempo enables the sound model to cope with issues of scale. Every extra synthesized voice requires more DSP resources, so it is not reasonable to dedicate a synthesized voice to every communicating participant when DSP resources are finite but participant-number (in principle) is unlimited.
- The number of participants who are the subject of awareness from other participants is calculated. The greater this number, the faster the rate of a pulsing sound. In this way, if all participants are concentrating on just a few others (the performers say), the pulse-tempo is slow. At the other extreme, if there is no single subject for attention and participants are showing a considerable degree of awareness of each other, the pulse-tempo is high.
- The mean average of the participants' displacements is computed. The greater this statistic, the greater the high frequency content of the pulse. Accordingly, if participants are relatively immobile, we hear a mellow pulse. This becomes sharper, the more movement there is.
- The number of mutually aware groupings of participants is calculated. The greater this statistic, the higher the pitch of a background drone. Thus, if all participants maintain a mutual awareness of each other, we hear a low drone. If the population of the electronic arena splits off into multiple groupings, we hear a higher pitched drone.

- The average separation between groups is calculated. The greater this statistic, the higher the pitch of the pulse. In this fashion, a population composed of groups who are closely packed will be sonified with a low pitched pulse. A population with highly separated groups yields a high pitched pulse.
- The scattering of the participants is calculated independent of any assessment of group membership. That is, the overall distribution in space of individuals is measured by calculating the standard deviation in their coordinates. This is sonified by splitting the pulse into two and panning one stream to the left and one to the right. The greater the scattering statistic, the greater the time interval between the two pulse streams. Using time interval and stereo location in this way gives a perceptually salient representation of scattering. The pulse is heard as more 'spread out' with the longer delays between the two streams. This, in initial informal listening trials, was more suggestive than using spatial location alone.

It should be clear from the above description how we are commonly concerned to sonify calculated statistics based on ongoing population data, rather than sonify the positions (say) of participants directly. Such direct methods may not scale well given limited DSP resources and, anyway, as we have argued, we wished to complement the presentation of raw (or 'near-raw') data in the visualization with statistical data in the sonification.
It is also worth noting how, under many circumstances, the data dimensions (and hence the sound parameters) may well covary in a correlated manner. A greater average distance between groups will probably be correlated with a greater scattering between individuals. That is, high pitched pulsing will often be associated with long intervals between two pulse streams. But not always. Indeed, we imagine that the breakdown of this 'standard' correlation would be especially salient perceptually. This should alert direction and production personnel to an unusual configuration of participants within the electronic arena. In this way, the exceptional will be signalled by unusual behavior in the sound model.

4.4.2. A Techno Musical Sound Model

We have also briefly explored how participant activity can sonified using more conventional music means. The pulse/voice model yields a set of sounds which many would judge 'unmusical'. Our second model uses some melody and rhythm lines typical of 'techno' dance music to sonify activity. For example, the number of participants who are the subject of awareness of others is reflected in the key of the melody line. The number of people communicating with each other is reflected in the volume of the hi-hat. The standard deviation of the population's positions is sonified by varying the pitch of the percussion. We offer this model principally as a demonstration that more conventional musical genres can provide the basis of sound models. We also offer it as proof that participant activity can drive the live control of music—something which might be interesting as an activity in an electronic arena in its own right (a matter we return to in the future work section of this chapter). However, we believe that conventional musical models might be overly distracting in the settings we principally intend for our sonification work—the support of production and direction. Musical models are easy to listen to *as music.* It is easy to miss-hear the variations in them as existing for musical reasons, rather than in terms of the data that are being represented. Additionally, musical material is likely to interfere with any other musical material that might be relevant to the electronic arena (e.g. broadcast musical content). Finally, techno music varied by the statistical analysis of participant activity can be extraordinarily annoying!

4.4.3. A Granular Synthesis Sound Model

Granular synthesis (GS) is a sound synthesis technique which employs a vast number of small sonic elements ('grains') to achieve a great variety of overall percepts (Gabor, 1947; Roads, 1987). Typically, quite simple grains are used (e.g. sine waves in gaussian amplitude envelopes lasting only a few milliseconds each) but in their combination much variety can be produced (e.g. 'roaring', 'tinkling', 'noisy' sounds as well as 'pure' tones). As many thousands of grains are used in the synthesis of sounds, the user of GS techniques almost invariably controls the overall statistical distributions of the grains and time-varying tendencies in the distributions rather than the character of individual grains. In many respects, GS has a 'natural' affinity with the sonification of large-scale data sources, as GS depends for its effects on the emergent perception of multiple individual sounds.
Our exploration of GS for the sonification of participant activity in electronic arenas is at a preliminary stage. We have explored a number of data dimension to GS parameter mappings. For example, the number of participants who are the subject of awareness of others can be represented in terms of the density in the time domain of grains. The number of inhabitants communicating can be represented by the distribution of grains in the frequency domain (with an appropriate GS sound model, this gives the impression of more 'talk' albeit in an abstract way). And so forth.

4.5. Experimental Evaluation

To subject our sonification work to more formal empirical evaluation, we conducted a psychophysical appraisal of the discriminability of the pulse/voice model as this is the most developed, stable and practically viable of our models. It also instantiates our sound model design criteria most clearly. As such, testing this model provides the fairest and most rigorous test of our sound model design principles. Our evaluation strategy is to first test the perceptual discriminability of the elements of the model and secondarily test the model in tasks that we imagine are more realistic to the intended application area. The justification for this is the view that if the model does not yield adequately discriminable sounds in psychophysical terms then it is unlikely to be practically usable. At the time of writing, we have completed enough psychophysical testing to feel entitled to proceed to the second stage of the more work-like, practical evaluation, which will take place early in Year 3 of eRENA. It is these psychophysical tests which we report here.

4.5.1. Experimental Design

Twenty-eight audio files in .aiff format (44.1 kHz, stereo) were prepared. These files were organized in pairs. In each pair, one sonification parameter was played out for a high value and a low value, respectively, while the other parameters were kept at a constant value. Each parameter was represented in two pairs, one with the low and high values being equal to the minimum and maximum values possible, and the other with the low and high values being equal to 1/3 and 2/3 of the maximum possible value. The length of the sound samples was approximately 10 s each. The presentation of the sounds was done as follows: All sound pairs were played in random order, with the ‘high value’ or ‘low value’ sample randomly chosen as the first to be played of the pair. Then the same pairs were played again, in random order, with the ‘low value’ and ‘high value’ samples of each pair in the opposite order from the first presentation.

The subjects were students at the Royal Institute of Technology, Stockholm University, and the Graphical Institute, Stockholm, who received course credit in exchange for their participation. The total number of subjects was 19, of which 15 were male. The median age was 24. 12 of the subjects played or had played an instrument regularly. No subjects reported any hearing deficits. Subjects were seated by an SGI Octane on which the test was performed. They were equipped with headphones, which we had pre-set to a suitable volume. The presentation of the experiment was automated using a Tcl script, which supplied the subjects with the necessary instructions, as shown in Figure 4.1, and then let them supply their judgements of which parameters they heard. Their responses were logged to a file.

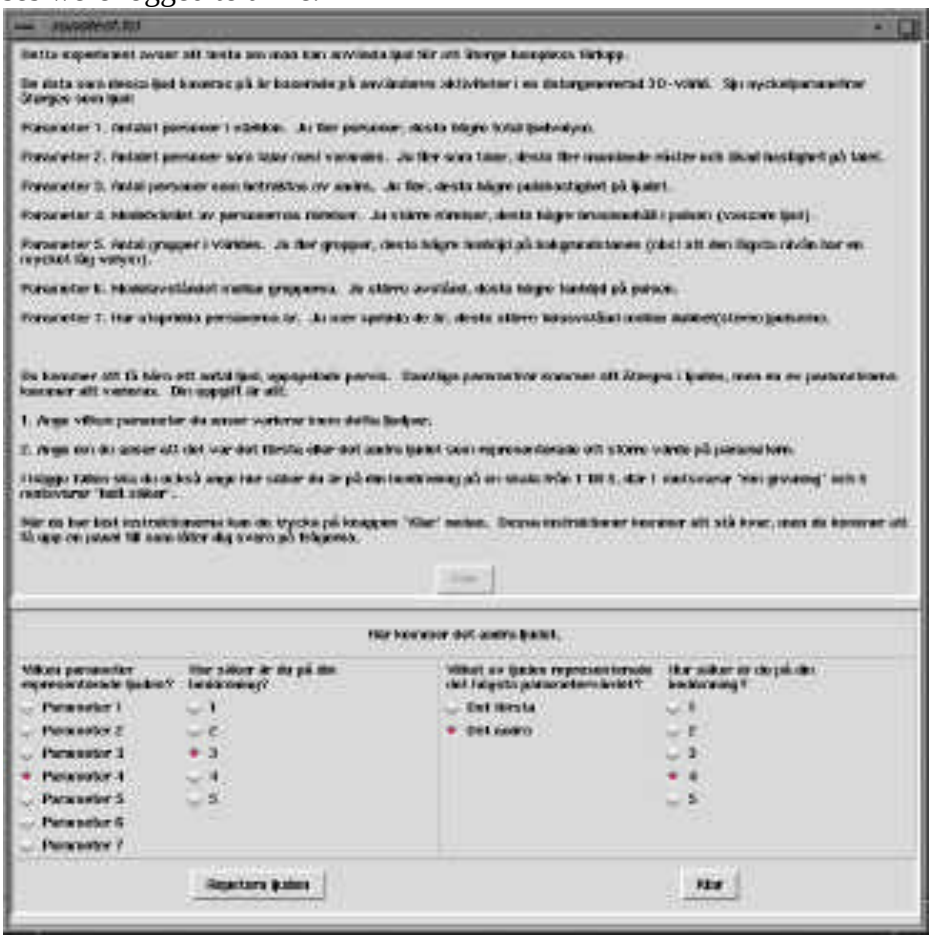

Figure 4.1: Instructions to subjects and response form.

These are the instructions translated into English:
“This experiment aims to test if one can use sound to represent complex processes.
The data that these sounds are based on are based on user activities in a computer-generated 3D world. Seven key parameters are represented by sounds:
Parameter 1. The number of persons in the environment. The more persons, the higher total sound volume.
Parameter 2. The number of persons talking to each other. The more speakers, the more mumbling voices and increased speed of speech.
Parameter 3. The number of persons watched by others. The more, the higher the pulse rate of the sound.

Parameter 4. The average of the person's movements. The larger the movements, the higher the noise content of the pulse (sharper sound).
Parameter 5. The number of groups in the world. The more groups, the higher the pitch of the background tone (n.b.! the lowest level has a very low volume).
Parameter 6. The average distance between groups. The larger the distance, the higher the pitch of the pulse.
Parameter 7. How scattered the persons are. The more scattered they are, the larger the time interval between the double (stereo) pulses.
You will hear a number of sounds, played in pairs. All parameters will be represented in the sounds, but one of the parameters will be varied. Your task is to:
1. Indicate which parameter you consider to be changing within this sound pair.
2. Indicate whether you consider that it was the first or the second sound that represented a higher value for the parameter.
In both cases you will also indicate how sure you are of your judgement, on a scale from 1 to 5, where 1 corresponds to 'pure guess' and 5 corresponds to 'completely sure'. When you have read the instructions you can press the button 'Ready' below. These instructions will remain, but you will get a panel that lets you reply to the questions."
The subject only received this verbal description of the sounds they were to hear, which may have affected the responses in the cases if the matching of a description to a sound was not obvious to the subjects, a point we shall return to. The subjects had the possibility to repeat a sound pair they were unsure of and the number of repetitions was recorded in the log file.

4.5.2. Data Analysis

The subjects clearly were able to tell the parameter sounds apart with better than chance results. Overall the subjects scored 77% correct answers. As shown in Table 4.1, the sound of parameter 4 was the most difficult to identify and the sound of parameter 5 the easiest. A more careful analysis, displayed in Table 4.2, shows that the sound of parameter 4 was often mistaken for the sound of parameter 6, as well as any of the other sounds; the sound of parameter 3 was confused with the sound of parameter 7, but almost never with any of the other sounds. We discuss below whether this is due to the sounds in themselves being hard to tell apart or if it just was difficult to match them with the given description.

Parameter	Correctly identified
1	50 (66%)
2	66 (87%)
3	61 (80%)
4	45 (59%)
5	72 (95%)
6	60 (79%)
7	58 (76%)

Table 4.1: Overall correct response rates by parameter.

Guessed parameter / **Actual parameter**							
1							
2							
3							
4							
5							
6							
7							

Table 4.2: Confusion matrix showing overall numbers of errors by actual parameter and subject response.

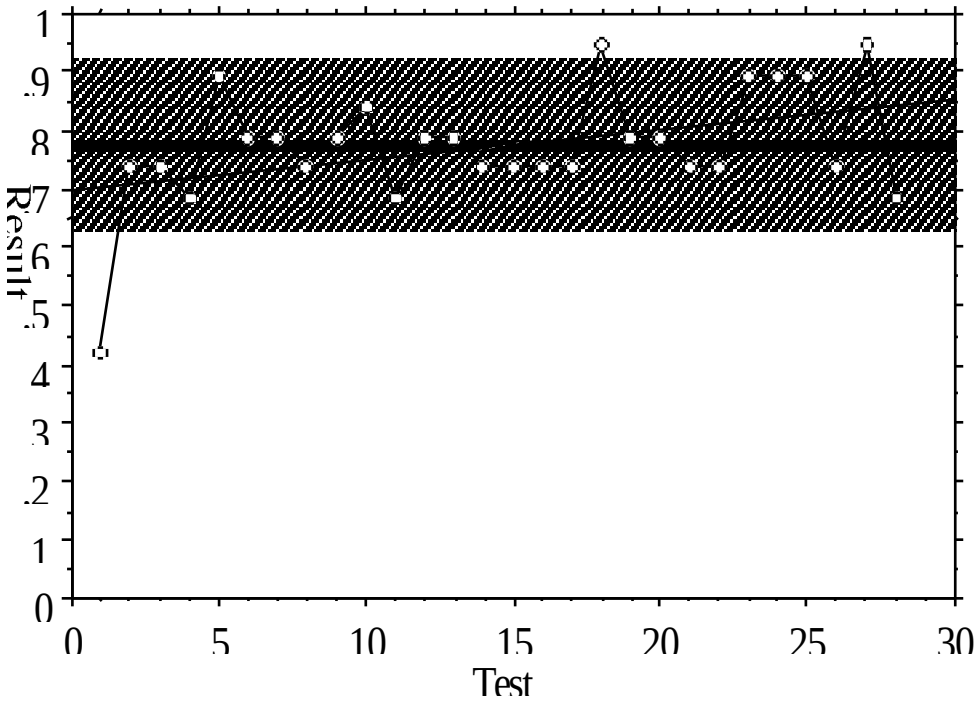

Figure 4.2: Proportion of correct responses by test item order. Regression line and 95% confidence intervals are also plotted.

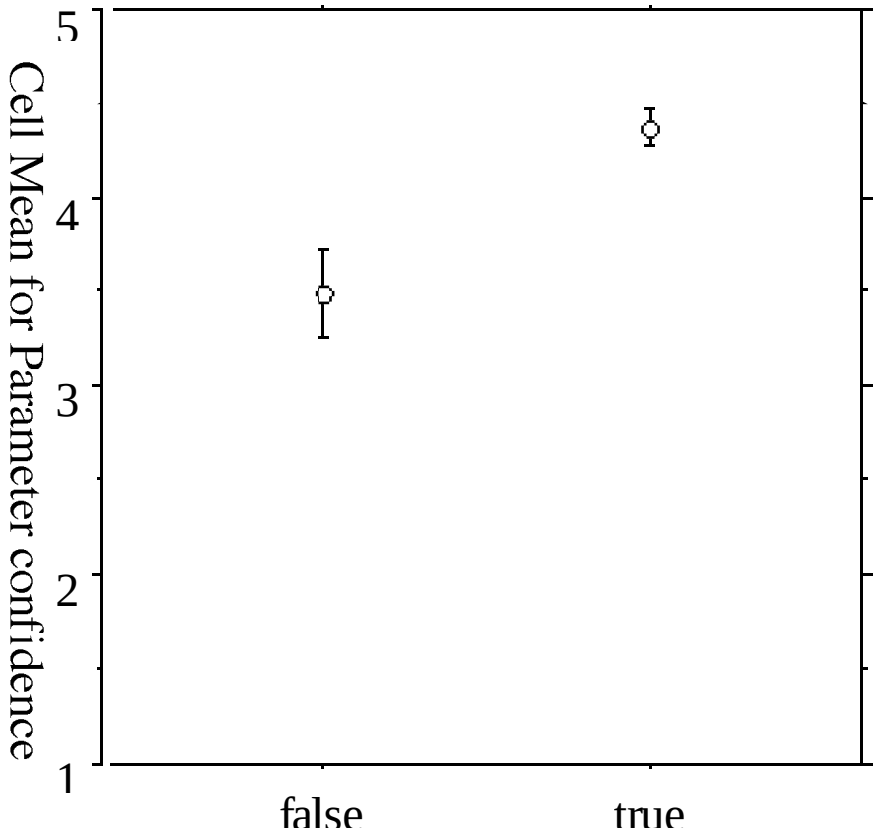

Figure 4.3: Mean and 95% confidence intervals for confidence ratings given to true and false responses.

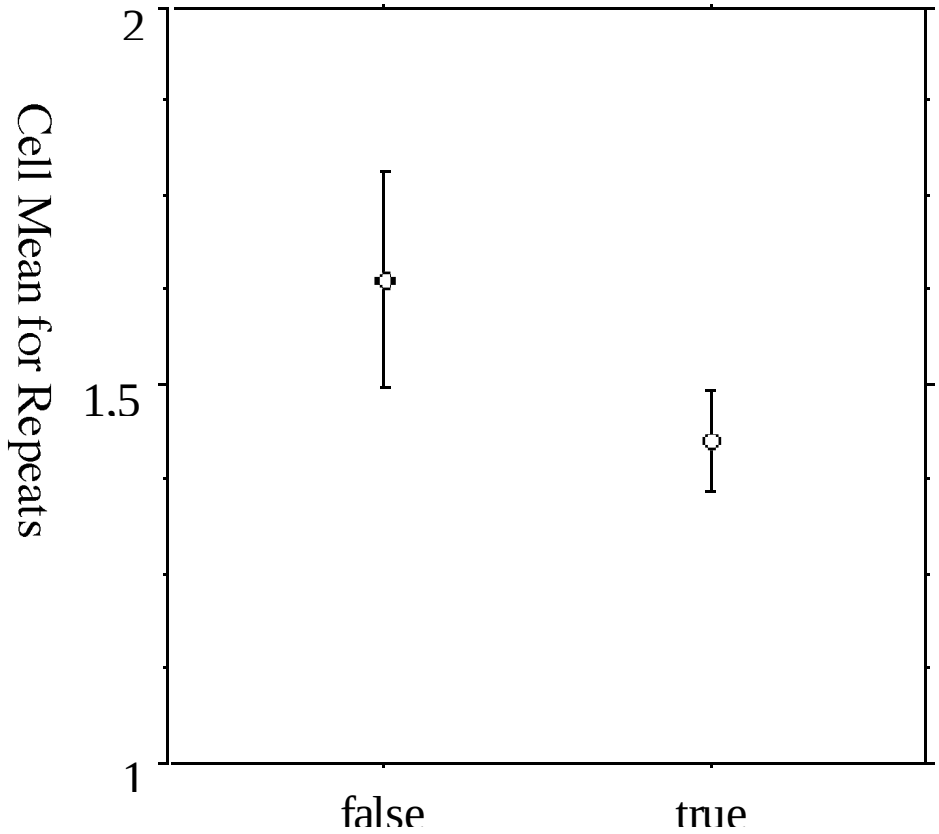

Figure 4.4: Mean and standard deviation for the number of times a stimulus was repeated by the subject for true and false responses.

Many subjects commented that it took them a few attempts to understand how the sonification worked, but the data indicate that they got up to their maximal performance very quickly, as shown in Figure 4.2. With a correct identification of a parameter as 1, and an incorrect as 0, we can plot the average results of the users over the course of the experimental run. We can see that most of the subjects misidentified the first sound they heard, but immediately after that they perform at approximately the same level for the rest of the run. There is a slight but statistically significant (at the 95% level) positive trend, so *some* further learning did occur over the course of the experiment.
In passing we can note that was a strong correlation between subjects guessing wrong and them being hesitant about their judgement, as indicated by the confidence ratings in Figure 4.3 and the number of times they repeated the stimulus plotted in Figure 4.4.

4.6. Conclusions

Our experimental results show that untrained subjects can quickly learn to reliably interpret the sonification of the data dimensions we calculated to represent the activity of participants in an electronic arena. Indeed, the majority of this learning is accomplished in one trial (see Figure 4.2). This strongly testifies to the appropriateness of the design of the pulse/voice sound model from the point of view of the perceptual discriminability between its dimensions of variation. Within single dimensions, high discriminability was also found for stimulus pairs that spanned the whole sound parameter range as well as for pairs that spanned a third of it. We believe, therefore, that the pulse/voice model can form the basis of future developments of a sonification model for application in electronic arenas and that, indirectly, our sound model design principles have been experimentally validated.
However, some of the details of the model are in need of improvement. The above results indicate confusions between parameter 4 (the pitch of the pulse) and parameter 6 (the timbre of the pulse). This is understandable from a psychophysical point of view. Physically, increasing the pitch of a stimulus will increase the high frequency content of the display. Our results suggest that either the pulse tone is not strongly 'pitched' enough (so that such changes are uniquely heard as timbral changes) or that the timbral changes are not currently dramatic enough. While this psychophysical interpretation of the confusion of parameters seems to us to be highly plausible, it is possible that some subjects may have experienced confusion in mapping the sonic changes to the underlying description given in the instructions. Some aspects of the descriptions are eminently discriminable semantically: parameter 4 is described as denoting 'the number of persons watched by others' and parameter 6 as 'the average distance between groups'. Other aspects may not have been so clear. In judgments concerning parameter 4, subjects have to be attuned to the "noise content" or "sharpness of pulse", while parameter 6 refers to "pitch". These terms could be semantically confusable for subjects as well as referring to perceptually confusable variations in stimulus material.
Similarly, there is confusion between the two parameters that influence the temporal distribution of sonic events. Parameter 3 affects the rate of the pulse. Parameter 7 affects the delay between two spatially separated streams of pulses. As the hit rate for both parameters is well above chance in spite of such confusions, we feel entitled to conclude that we need to exaggerate the perceptual difference between the two parameters rather than to fundamentally redesign the sound model. There are many possibilities here. For example, the perceptual difference between the two sound streams could be exaggerated, perhaps with this difference increasing as time delay between the streams increases. This would add another 'perceptual dimension' to the change in parameter 7,

hopefully increasing the discriminability of changes in that parameter from changes in parameter 3. Again we feel that the psychophysical observation that both parameters 3 and 7 affect the temporal distribution of sonic events encourages a psychophysical interpretation of the confusion data, though it is also possible that some subjects found the descriptions hard to understand. While parameters 3 and 7 refer to quite different sources of data being sonified, subjects are required to make judgments over whether they are hearing "a high pulse rate" or "short time interval between two stereo pulses".
It should also be observed that parameter 1, which involves changes in overall sound amplitude, had a hit-rate which could be improved. While 66% is still above chance, it is clear that a number of subjects had difficulty discriminating amplitude changes in our forced choice experimental design. A post hoc analysis of the data reveals that performance on amplitude discrimination test items is considerably better if such items occur later in test order than earlier. That is, being sensitive to amplitude changes is the one parameter that needs more significant learning. We do not believe that this result negates the use of amplitude variation as a parameter in a sound model, though it must be granted that amplitude levels and variation will have to be carefully calibrated in any real practical application setting (e.g. becoming attuned to amplitude changes might be further disrupted if the users were to decide to turn the auditory display down if it became intrusive for practical purposes!).
In general, we conclude for our first phases of experimental study that the pulse/voice sound model is appropriate and is based on valid design principles for us to proceed, as we shall in Year 3 of eRENA, to examining it and related models in more work-like settings with more realistic experimental tasks.

4.7. Future Work

Let us conclude this chapter with some reflections on our experience of developing sonifications for the support of production and direction of events in electronic arenas. As in our visualization work, we are exploring the use of representations of participant activity as a resource for production and direction. This is a consistent feature of our research strategy in eRENA—using activity data to inform production and direction work, rather than, say, geometrical features of the virtual environment (matters which have been well explored in the work of others, see Chapter 3). However, in our sonification research, we have extended these principles by computing simple population statistics and representing them in sound. This seems to be a promising way of building complementary cross-modal displays. It must be admitted that our research will only be evaluated in a work-like simulation in Year 3, though preliminary evaluations of the psychophysical basis of our design principles for sound models are favorable.
We have pointed to various ways, suggested by the results of experimental evaluation, in which our sound model can be improved. It must also be noted that there are other features of the organization of sound which we could exploit if further data dimensions need to be sonified or more redundancy needs to be introduced into the sound model to increase the discriminability of variations in data dimensions. For example, we have not exploited variations in the dynamics of sound through, say, having repeating rhythmic patters marked by different accented elements. This might be especially salient when rhythmic musical material is used for sonification purposes as in the techno model we have sketched. The more thorough exploration of alternatives to the pulse/voice model will be required to properly understand the different sound parameters that are appropriate for use for the purposes of sonifying participant activity in electronic arenas.

An obvious problem with exploratory research of the sort we are engaged with here are the degrees of experimental freedom which are open to the researcher. There are an indefinitely large number of sound models, controllable in an indefinitely large number of ways, which can be associated with an indefinitely large number of complementary visualizations. Our strategy to make this manageable is to focus on just three mutually different sound models which embody highly contrasting methods of sound synthesis but to confine ourselves to a relatively small number of data dimensions. We further constrain the design space by working with quite restrictive criteria for sound models. Though restrictive, we believe these criteria are reasonable from both psychophysical and practical points of view, and leave a number of avenues open for the creative selection of sound models.

We also work with an application focus to further constrain the design space: providing sonifications to support the production and direction of events in electronic arenas. However, a number of extensions of our work into other application areas suggest themselves. First, while we intend auditory displays for the use of production personnel, there is no reason why sound controlled in a similar fashion might not be made available to other participants in events in electronic arenas. For example, inhabitants might equally benefit from hearing the activity in the electronic arena, whether this is to serve as a practical resource for their navigation (or other kinds of behavior) or as 'world sound' intended to be of aesthetic interest for them. Indeed, one can envisage events in electronic arenas that have the collective synthesis of sound as one of the main aspects of them. Techniques such as those we have explored could well be used in such settings.

Currently, our auditory displays are based on a 'bottom-up' approach of taking real-time data from every participant and computing population level statistics that are then sonified. A promising extension of this is to take parameters underlying the crowd simulation models in Workpackage 5 (see Deliverable D5.4) as sources of value for the sound model parameters. In this way, different parameterizations of the crowd models could yield differently sounding crowds. The 'sound of the crowd' could be further controlled by using changes in the 'interest points' and 'action points' that control the crowd behavior as source for some sound parameters.

Sound models could be abstract or more literal. Our pulse/voice sound model gives an indication of how sounds with vocal qualities can be synthesized with finite DSP resources. Extensions of this could sonify a 'chattering crowd'. We have also given an indication of how sound synthesis methods like granular synthesis (GS) seems idiomatic for the representation of large-scale, collective events. It is eminently feasible to use GS techniques to synthesize quite realistic crowd sound including sounds such as 'rustling', 'chattering' and 'roaring' - all of which, given an appropriate context, could be useful for giving a sonic representation to collective activity.

We have currently investigated strategies for developing complementary and mutually informing auditory and visual displays. To date, we have studied non-manipulable sonifications to complement manipulable visualizations. However, other relationships could be easily explored. For example, a degree of manipulability could be introduced to the auditory display. Individual sound components could be muted or some parameters momentarily fixed to a constant value so that other aspects of the model could be more carefully scrutinized. Another possibility to support more detailed listening would be to introduce facilities for 'jumping offline' by, for example, capturing a sample of data and its sonification. The sample could be replayed or looped at varying speeds. While we primarily intend sonifications of participant data to support users in obtaining a non-intrusive overview of events in an electronic arena, it is not hard to think of interaction techniques to extend the interactivity of our auditory displays.

More fully interactive auditory displays would allow other relationships between sound and vision, and other applications, to be explored for electronic arenas. For example, a multi-dimensional visualization could form the interface to complex sound synthesis methods for music performance or composition purposes (for a review of some approaches to this, see Pressing, 1997). In Deliverable D2.2 from Year 1 of eRENA, we described the application SO2 in which a 2D spatial interface is provided to an algorithmic layer which generates streams of parameter values for methods of physical modeling sound synthesis. The user interacts with SO2 by providing a location in a 2D-screen space using gestures with the mouse. Our visualization work suggests another interaction paradigm. The user (or a group of users) could interact with a 'population' of interface objects and various features of their distribution and dynamical behavior could drive a sound model. These objects might also be given a computer graphical rendering. In this way, the real-time generation of visual and sonic material could be controlled in an integrated flexible way. Applications of this sort would seem to be ideal for implementation using the framework developed at the ZKM and described in Chapter 2 of this Deliverable and are intended to form a focus for collaboration in Year 3 of eRENA with practical demonstration and testing forming part of these two partners' contribution to the planned workshop on sound in electronic arenas, December 1999.

Chapter Five
Round Table: A Physical Interface for Virtual Camera Deployment in Electronic Arenas

Michael Hoch
Zentrum für Kunst und Medientechnologie (ZKM), Karlsruhe, Germany

Kai-Mikael Jää-Aro and John Bowers
Royal Institute of Technology (KTH), Stockholm, Sweden

5.1. Introduction

In Chapter 3 of this deliverable, we described the approach we are developing in the eRENA project to supporting event management in electronic arenas via a notion of activity oriented virtual camera deployment and control. That is, it is the activity of participants in an electronic arena that serves as a resource to guide the deployment, direction and control of the views that are made available for broadcast or other forms of dissemination. We have proposed various means by which indicators of the activity of participants to an event may be determined. We have presented a prototype system, SVEA, in which measures of activity and participants' awareness are visualised and sonified (see Chapters 3 and 4) so that virtual cameras might be deployed to capture the action. We have also presented a number of algorithms for computing near-optimal camera shots of various kinds that could serve as initial deployments amenable to manual refinement.

It has been an important emphasis of all this work (and of the work reported in Chapters 1 and 2) that technical systems be proposed and developed which are capable of real-time operation in settings which have a considerable degree of unpredictability (e.g. because performer-improvisation or mass-public participation is emphasised). This has given all the work presented in this deliverable a distinctiveness from animation and other computer graphical techniques which tend to envisage non-real-time applications (e.g. those in the film industry).

The emphasis on our production support technologies as being feasible in real-time operation must be followed through into a consideration of usability issues. It is considerations of actual usability that motivate the concern to develop an environment in which programs do not require compilation (see Chapter 2). The SVEA application is able to receive and visualise/sonify a real-time stream of position, orientation and activity data from an electronic arena. Computationally expensive algorithms for the optimisation of camera positions or paths, or those that require pre-computation, have been avoided (see Chapter 3). However, it is important that we complement these considerations with a sensitivity for what is and is not possible for end-users to do in the real-time hurly-burly of live events. No matter how sensitive technologies are to real-time computational considerations, they may well fail if it is not practically possible for users to fold them into *their* real-time: the real-time of the co-operative work of production and direction of

events. An application, for example, which would require time consuming activity at the interface selecting from hard to access menus or with physical interfaces which could not be used dextrously would be unlikely to be acceptable.

This chapter, then, is concerned with investigating interaction techniques and devices which are appropriate for real-time operation of the kinds of production software we have been developing. Specifically, we describe our initial development work of a novel *physical interface* (embedded within a *room-sized environment*) for the control of virtual cameras in applications in electronic arenas. Several usability issues have prompted this proposed solution, including the following.

1. Interaction using conventional desktop input devices such as mice, joysticks and keyboards is often too slow when time-critical selections are required (e.g. precisely timed cuts between camera views made by a director or deployments of cameras to where the action is at the very moment it is occurring). To move icons or make selections with a mouse requires the user to first grasp the mouse, then make a controlled movement on screen to the target (icon or menu), engage with the target, and then execute the appropriate function. It is reasonable to believe—and is often claimed—that with an appropriately designed physical interface, engaging with the target (phicon or push-button) can be accomplished with less preparatory movement.
2. Image direction in the settings of interest to us is a co-operative activity, where multiple users (directors, camera operators and other production members) need to sustain awareness of each other's gestures around shared artifacts—such forms of mutual awareness being very commonly documented as an essential feature of cooperative work in time-critical settings (see, e.g., Martin, Bowers and Wastell, 1997, and the results of the field study of *OOTW* presented in Deliverable D7.a1). This would tend to speak against environments where each participant would only have access to events in an electronic arena their own monitor display. Environments where views of an electronic arena can be shared and where mutual access to each other's activity with respect to these views can be naturally picked up would seem to be worth exploring.
3. Real-world space needs to be recognised and reserved for participants to bring freely whatever real-world documents and other artifacts they wish allowing interaction with these to be interleaved with technically-mediated interaction with production support applications or virtual environment exploration. Field study in both inhabited television and media art settings in eRENA has revealed the obduracy of paper notes, running orders, and various bits and pieces of equipment which need to be worked with alongside any production software one might wish to develop. These phenomena tend to speak against fully immersive solutions or the hope that everything that a production crew would ever need could be rendered on-screen.

For all these reasons, we are investigating physical interfaces and artifacts, on a human-scale, to be sited within room-sized environments, as the appropriate way to make our technologies for production support in electronic arenas available to users. The current chapter gives details of our progress so far in fleshing out this design image. We start with a brief comparative review of well-known work on mixed reality shared environments and interfaces. We then detail our own physical interface to SVEA, including some of the novel techniques we have added in to the

application to take advantage of physical interaction methods. We close with an appraisal of the current status of our work and a look to future developments.

5.2. Mixed Reality / Shared Environments

Several approaches to integrate physical and virtual space in a shared environment have been proposed, for example, DigitalDesk (Wellner, 1991), Bricks (Fitzmaurice, Ishii and Buxton, 1995) and phicons (Ishii and Ulmer, 1997). Based on these foundations some applications have been shown to successfully integrate physical interaction handlers and virtual environments or tasks, as in the System BUILD-IT (Rauterberg et al., 1998), where engineers are supported in designing assembly lines and building plants, or in URP (Underkoffler and Ishii, 1999) where a physical interface is used for urban planning, or the concept of 'Embodied User Interfaces' (Fishkin, Moran and Harrison, 1998) where the user physically manipulates a computational device.

In the table environment of Rauterberg et al. a menu area is proposed for object selection that, thereafter, can be placed on the virtual floor plan by moving the interaction handler. This approach uses the physical object as a general interaction device. The physical objects that are used in Underkoffler and Ishii for the urban planning example are mostly used in a less generic but more specific way which lowers the chances of errors due to user input, e.g. a building phicon would less likely be used as something else than a generic brick object. Another approach is reported in Ullmer, Ishii and Glas (1998) where physical objects, the so called 'mediaBlocks', are used as digital containers that allow for physical manipulation outside of the original interaction area.

The input devices proposed in this chapter extend these approaches in a number of ways. First, we introduce a context sensitive functionality to the physical objects a user interacts with. That is, the exact significance of an action on a physical object can change in relation to the context in which the action is performed. This enables us to support several different kinds of user action without proliferating the number of phicons that need to be used and identified. Second, we propose a setup that combines physical interaction with abstract visualisation in an application that is not concerned with the off-line design of an environment, but real-time intervention in an electronic arena. This combination of physical interface with abstract visualisation and real-time consequences of interaction gives our work uniqueness within exploration of physical interfaces. Finally, we emphasise the overall working ecology in which the physical interface we have prototyped is designed to fit. We imagine a room-sized co-operative environment where physical interfaces might enhance and add to traditional interfaces and work activity. This concern for realistic co-operative working environments is rarely emphasised in the design-led demonstrations of physical interfaces and tangible bits that are commonly reported.

5.3. Round Table with Interaction Blocks

A round table with a projection screen in the middle is used to display a map of the electronic arena (see Figure 5.1). The image on the table-screen is rear-projected—that is, projected from underneath the table using a projector and a mirror. The projection screen is approximately 80cm across with a table height of approximately 95cm. Physical objects are placed upon the table-top projection screen to deploy cameras, select cameras for transmission (TX in broadcasting terminology), enable zooming of the display and the other operations we shall shortly describe.

Figure 5.1: Table with rear projection

On a second projection screen next to the table (to the rear of Figure 5.1), a 3D rendered scene can be displayed from the perspective of the deployed camera. Alternatively, the camera view, as well as the TX view, can be shown on additional monitors in a room-sized environment.

Figure 5.2: Top view of table with floor plan, abstract visualisation of avatars and interaction blocks

A pole mounted on the table holds a real camera with infrared light. It is used for tracking blocks that can be placed on the table screen (see Figure 5.2). These can signify the position of virtual cameras by means of positioning the interaction blocks on a representation of the virtual scene. In Figure 5.2, a SVEA visualisation of participant activity is shown with participants depicted as shaded triangles in the manner described in Chapter 3. To the right a larger triangular

block can be see—the phicon representing a virtual camera. Two other phicons are also present in Figure 5.2: two small objects that appear circular from above. These are used to make selections in the display in a manner that we describe below. The selections they have made are indicated by the darker highlighting of the triangles proximate to each of them.

5.4. Vision Based Tracking

For tracking the movements of the interaction blocks on the round table, we used mTrack—the same vision based analysis system that was used in the performances described in Chapter 1 of this deliverable. Analysis is based on infra-red illumination, luminance-level segmentation, and blob analysis. The system is divided up into the recognition part consisting of the image processing system, a server program, and the library front-end with the application program (see Hoch, 1997, 1998). The image processing system tracks the blocks via a relatively low-cost off-the-shelf infra-red camera set-up that is mounted on the pole of the table. After initialization the system continuously sends position data or other calculated information concerning the segmented objects to the server program. The server program connects an application with the image processing system. It updates the current states by an event driven loop. Upon request it sends data to the application continuously. The information that is currently extracted is position data, shape information of all blocks, as well as orientation information for a triangular shaped block. For robust segmentation of the blocks on the projection table, we use retroreflective colour attached to each block (made available by 3M, Neuss, Germany) and an infrared filter on the camera to eliminate visible light. For each block present in the scene, this enables the detection of a brighter reflection spot than would be possible with unfiltered room lighting incident upon less reflective surfaces. This greatly facilitates tracking by enhancing the contrast in the image that is input to mTrack's analysis routines (which are described in more detail in Chapter 1).

5.5. Interaction

We currently use four different physical objects for interactional purposes (see Figure 5.3). The isosceles triangular shaped object is a *camera* phicon and is used to deploy virtual cameras in the electronic arena. As in the sense given to such shapes in the SVEA visualisations (see Chapter 3), the position of the camera is taken to be at the 'sharp' apex of the triangle with its direction of view 'away' from this point along the shape's axis of symmetry. The triangle then can be naturally interpreted as an arrow-like pointer in the direction of the camera's view. Once a new camera phicon is detected by mTrack, a virtual camera is assigned to the location indicated and pointing in the direction suggested. The view onto the electronic arena from this virtual camera can be selected for transmission (TX) by placing a small round *camera selector* phicon into the non-reflective hole in the middle of the triangle.

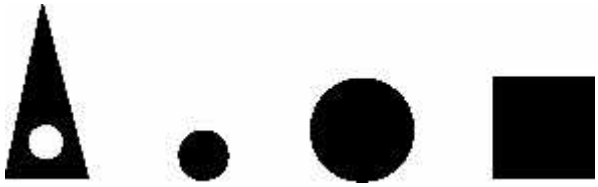

Figure 5.3: Shapes of detected phicons: camera, camera selector, probe, and zoom (showing their relative sizes—the camera phicon being approximately 8cm long)

Using the camera selector phicon is proposed as an intuitive and simple way for selecting cameras. We preferred this technique over using switches and additional lights mounted on the phicons themselves as this would have required more complicated sensing devices and battery powered phicons.

The *probe* phicon is used to select a group of avatars in the projected visualisation. Rather than attempt—using phicons—to replicate the mouse gesture of clicking and dragging used to select groups in the screen-based SVEA application described in Chapter 3, we decided to exploit and extend the awareness model underlying the visualisation to enable context-sensitive selections to be made. Let us explain this in more depth. Imagine an object in the electronic arena at the position corresponding to a probe placed on the projected visualisation. Assign focus and nimbus (see Chapter 3) to this object just like other objects and avatars in the electronic environment. Just as we computed the awareness participants in the environment can have of each other (remember this is what is signified by the basic shading of the triangles representing participants), it is possible to determine the awareness the probe-object would have of avatars in proximity to it. The avatars with an awareness level above a given threshold can be returned as a selected group. In this way, the probe can be used to select the group of avatars that the probe would be aware of from the spot where it is deployed. When a new probe phicon is detected by mTrack, the group of triangular avatars identified in this way is highlighted by darkening their colour (see Figure 5.2 where two groups have been selected by two probes).

We describe this use of the probe phicon as being 'context-sensitive' because exactly which avatars it selects, how many and in which configuration, is dependent upon the avatars' orientations and proximity with respect to the probe (at least this is so in the implementation of the Spatial Model we have followed since Chapter 3). If the avatars are sparsely distributed in the environs of the probe, maybe only a few will be selected. If the avatars are more densely packed where the probe is deployed, very many may be selected. However, in both cases, the same basic gesture—placing a probe phicon upon the table-top projected visualisation—will be used to make the selection.

Although different methods are described in Chapter 3 for explicit group selection at the interface (i.e. mouse-dragging a selection-box), a virtual camera is algorithmically assigned to a group in the same fashion as soon as the group is identified. The possible algorithms for computing location and angle of view of this camera remain as described in Chapter 3.

Finally, the *zoom* phicon allows one to zoom into the visualisation in a similar context-sensitive fashion. The group of avatars is determined corresponding to those which an object placed in the electronic arena at the location corresponding to the zoom phicon would be aware of. The display is rescaled so that it shows this group plus an area around them. The extent of the extra surrounding area can be set as a preference (we have worked with displays which zoom to an area approximately twice the 'width' of the group, with the centre of gravity of the group at the centre of the zoomed display). This method also demonstrates our principle of introducing context-sensitivity into the interpretation of the deployment of phicons at a physical interface. The level of zoom of the visualisation is dependent on the number of avatars present in the area where the zoom phicon is placed. As described in Chapter 3, when we zoom the display, we do not scale the triangles representing participant-avatars. Deploying the zoom phicon enables the user to 'separate out' densely populated areas where otherwise many avatars might be shown on top of each other. Once the relations between avatars in an electronic arena has been clarified by

exploiting the zoom feature, camera phicons can then be positioned to get more appropriate views than would be possible with a uniform unzoomable display.

Figure 5.4: Zoomed-in visualisation with zoom phicon, two probes and camera phicon

5.6. Co-ordinating Multiple Virtual Cameras

In physical/real television or film, there is a finite limit to the number of cameras that can be used to capture a scene. When events in an electronic arena are depicted using virtual cameras, there is no in-principle limitation on the number available. Virtual cameras (like most things virtual) can be created on demand in a way that physical cameras cannot be. Furthermore, the analyses of network traffic during *OOTW* presented in Deliverable D7a.1 show that there is little penalty in system and networking resource terms in multiplying the number of virtual cameras *provided that those cameras are not actually graphically rendered in the electronic arena*. In the production of *OOTW*, though, it was chosen to limit the number of virtual cameras to four for various practical reasons. For one thing, the camera interface required one human operator per virtual camera. For another, there were physical restrictions on the number of inputs that could be accepted in the conventional television video-mixers which were used and which the director interacted with. Finally, four cameras each with a job to do within a working division of labour seemed about right to the director and producers of *OOTW* given the content of the show and the practical task of directing human operators (on these and related details, see Deliverable D7a.1).

However, if we anticipate the possibility of selecting and mixing cameras through software (as we do in the description of *Blink* in Chapter 6) and imagine that the sources that a director edits between may well be captured by cameras some of which are partly or wholly autonomous (i.e. algorithmically controlled), then some of these constraints may not apply to more fully developed electronic arenas.

In the round table production environment we have been discussing, design decisions have to be made, therefore, over issues to do with the relationship between inserting a new camera phicon into mTrack's field of view and the 'lifecycle' of a corresponding virtual camera. Does deploying a phicon create a new camera at the designated location? Or does it merely redeploy an available camera? Are there as many camera phicons as virtual cameras (and no more)? Or can virtual cameras be created without upper limit? Does the removal of a camera phicon cause the

corresponding virtual camera to pass out of existence? Or does it cause the virtual camera to return to some default behaviour?

Ultimately, we feel that such questions have to be answered with respect to particular applications. A priori, one can argue either way on a number of these issues. It could be that some events in an electronic arena involve action on such a mass distributed scale that it would be unreasonable to set an upper limit on camera number but necessary to extensively use autonomous cameras. It could be that more intimate events would be conducted like *OOTW* with a smaller number of cameras and much manual control. It could be that a production crew would find their work very hard to accomplish practically if they could not refer in traditional fashion to "Camera 1... Camera 2..." and so forth, and assign cameras to a fixed number of roles. Indeed, for aesthetic reasons, a director, designer, producer or artist may prefer even a single camera and make no cuts (perhaps in the name of a kind of 'fly on the wall' documentary direction style for electronic arenas!). Even in this extreme case, our round table production suite might be of use for the clues it would give for where the action is.

The fact that one can argue either way on these issues is further fuelled by the fact that *from a viewer's perspective* editing between multiple virtual cameras can be perceptually identical to following a single virtual camera which is capable of teleportation and changes in behaviour. This observation testifies to our point that whether one works with a system that has multiple virtual cameras, or just one, and what relationship the gestures of production staff have to the lifecycle of a camera, are largely matters to do with how best to facilitate the practical work of production of events for electronic arenas.

Our prototype, then, arbitrarily restricts the number of cameras to a user-preferred limit but does so without prejudice to alternative possibilities. Within this set of cameras, the default behaviour is an autonomous one. That is, if a camera is not deployed through activity at the round table, it maintains behaviour that is entirely algorithmically determined. This default behaviour is to rove the space, following the gradient of increasing awareness, while avoiding other cameras (i.e. the puppycam behaviour described in Chapter 3). The introduction of a camera phicon will 'claim' the next available camera. Which camera is 'next' is determined in a 'round robin' fashion. Thus, one will tend to deploy the camera that was least recently deployed. The round robin will pass over unavailable cameras. Cameras can be made unavailable by declaring them in advance to have fixed behaviour (e.g. to always stay in puppycam mode) or through selection for TX (thereby avoiding the transmitted view being suddenly and mistakenly cut to another location). The removal of a camera phicon from the round table will 'deassign' any associated virtual camera and return it to its default behaviour.

5.7. Updating the Relationships Between Visualisation, Physical Interaction and Virtual Cameras

On this scheme it is possible for a camera phicon to rest on the visualisation on the round table but to no longer have a virtual camera associated with it. This could occur if several deployments of camera phicons had occurred and the round robin method 'stole' an 'older' phicon's camera. This could lead to misunderstandings if users thought that triangular phicons always represented the presence of a camera in the electronic arena. For our round robin method of camera allocation, this would be an inappropriate 'user-model'. Rather, the camera phicons should be

regarded as *tools with which to deploy virtual cameras*, not representations of those cameras themselves.

In our current implementation of SVEA, we show the location of virtual cameras by means of small graphical depictions in the projected visualisation. In this way, we hope to make it plain to users where the virtual cameras are in the electronic arena and which camera phicons (still) have a virtual camera assigned to them. In Figure 5.5 a virtual camera is shown alongside the triangular phicon it is here associated with. Figure 5.2 shows a number of other cameras roving around the electronic arena not associated with any particular camera phicon.

We hope (though this is something we need to more formally investigate) that this variable relationship between virtual cameras and camera phicons will not be confusing *provided users entertain the 'tool' model rather than the 'representational' model of the relationship.* It is important to observe that similar issues arise for many attempts to physically interface to the virtual and are not specific to our application or our particular design decisions. Only in the limit case of a completely strict coupling between physical activity and consequences in the virtual world (and vice versa) would it be possible to think that a physical object could non-problematically represent a virtual one. As soon as the coupling is relaxed the relationship between the physical and the virtual has to be *achieved in users' practical understandings* rather than technically mandated (see Bowers, O'Brien and Pycock, 1999).

5.8. Current Status, Conclusions and Future Work

In Year 3 of eRENA, we will test a prototype round table in two settings: (i) a multi-camera direction application for electronic arenas, and (ii) an architectural application which allows participants to place building blocks on a floor plan to interactively modify the design and easily switch between viewpoints within the architectural visualisation. This second application will enable our physical interfacing techniques to be more readily compared with others in the literature (where architectural applications are prominent) while also testing the generality of our approach. Architectural applications are also of central relevance to eRENA as virtual architecture or set-design will be important to many electronic arenas. We plan user-testing at this stage as there are several features of our design which urgently require evaluation beyond our own appraisals. Most notably, we need to see whether the relationship between camera phicons and virtual cameras that we advocate above can be understood and practically acted upon by users. This is a matter of importance not just to our application but to others where there is less than a strict coupling between activity in the physical world and virtual world correlates. Evaluative feedback is also required on several other details in our design: for example, the context-sensitive probe and zoom concepts. We would also like to investigate whether concurrent sonifications of participant activity in setting (i) above would enrich or confuse a working environment. Finally, we anticipate that deploying cameras by moving physical objects to be faster than using a conventional GUI but this is something we should empirically examine (indeed, as we have both conventional and round table interfaces to SVEA, this can be tested directly).

We believe that our emphasis on combining physical interfaces and virtual displays within a shared environment designed to support co-operative work is novel, and that physical interfaces for virtual camera control in electronic arenas is a unique application area. Currently, we have built a functional prototype for deploying virtual cameras that shows the applicability of our

approach. We introduced a novel context-sensitive use of physical objects and presented novel selection and zoom operations using this technique. There is an intact chain of reasoning in our work from (i) the recognition of the need to support production staff in electronic arenas in finding 'where the action is' (a requirement revealed in the ethnographic study of inhabited television reported in Deliverable D7a.1) through (ii) the development of visualisations and sonifications of participant activity to serve as a resource for what we have been calling 'activity oriented camera control and deployment' (see Chapters 3 and 4) to (iii) the prototyping of a physical environment, and physical interfaces within it, which are sensitive to considerations of usability under the exigencies of real-time performance. What we do not yet have is a completed iteration in design which would take our prototypes back to potential users.

Nevertheless, in the round table environment, we do have an image of what a work setting might look like, where production and direction staff could work co-operatively and in interaction with algorithmic processes of various sorts in the real-time management of an event in an electronic arena. Put together with the more specific applications developed for inhabited television (reported in Deliverables D7a.1 and D7a.2, see also D5.3), and in the service of artistic performance (e.g. the work reported in Chapters 1 and 2 of this deliverable), eRENA is beginning to offer a varied toolkit for event management in electronic arenas.

Chapter Six
Blink:
Exploring and Generating Content for Electronic Arenas

John Bowers and Kai-Mikael Jää-Aro
Royal Institute of Technology (KTH), Stockholm, Sweden

6.1. Introduction

In this chapter, we describe *Blink*, a performance involving the improvised real-time construction of virtual environments which formed part of an evening called *Digital Clubbing* at the Nottingham Now98 arts festival, October 1998. Many elements of *Blink* exemplify the kinds of interactive technology that need to be developed for electronic arenas to support the construction of dynamic virtual environments and the co-ordination of multiple viewpoints upon them. Although *Blink* was performed on this occasion, like many of the experiments described in Chapter 1 of this deliverable, as a non-participatory event, there is no in-principle problem with extending its technologies to support audience participation. Again like several of the experiments in Chapter 1, *Blink* employed multiple screen projections to create an image-rich environment, rather than support immersive participation or inhabitation. In its concern for developing content (virtual environments and computer graphical constructions) for the event in real-time, for capturing such material from multiple points of view, and for distributing views to multiple screens and projection surfaces, *Blink* has many features of a nascent electronic arena. In addition, we developed software for *Blink* designed to interpret many of the dynamic features of the music for the *Digital Clubbing* (music by 4hero, who played live at The Bomb, the venue where the event was held, and Carl Craig, who also played live but via a link with his studio in Detroit, USA) in terms of parameters for the generation of computer graphical material. In this way, *Blink* is concerned with exploring potential relations between media, here between sound and vision.

In Deliverable D2.2 from Year 1 of eRENA, we describe *Lightwork* (Bowers, Hellström and Jää-Aro, 1998). As *Blink* extends many of the features of *Lightwork*, it is important that we summarise our earlier work here. In *Lightwork*, virtual environments were algorithmically generated in the real-time of performance. That is, a greater part of the goal of the piece was to explore the technical and aesthetic viability of interactively constructing virtual environments as the very topic of a performance work. To enable this, we pursued what we referred to as 'algorithmically mediated interaction'. That is, performer gesture does not generate virtual content directly object by object or image by image. Rather, performer gesture is analysed so as to be interpreted as supplying parameters for algorithms that generate world content. A variety of algorithms were developed for *Lightwork* to generate 'chambers' or 'immersive forms' through that the viewpoint moved, 'scaffolding' that filled the virtual environment with strongly angled forms to emphasise a sense of three dimensionality and depth to the image, and various other algorithms that added image and animated forms to the environment. In addition, the viewpoint

itself was animated along a non-linearly modulated circular path whose notional radius and 'bendiness' were also influenced by parameter values extracted from performer gesture.

One of the performers of *Lightwork* employed a series of footswitch controllers and a MIDI electronic wind instrument to generate the graphical material (as explained in Deliverable D2.2, one of the aims of *Lightwork* was to investigate 'cross-modal' devices for the control of the sound and graphical material: e.g. a musical instrument was used to generate graphical material). The playing of the electronic wind instrument was analysed for its pitch, MIDI velocity and timings between notes in terms of three moving time windows (the last 20, 100 and 500 notes). Statistics for the mean level and range of values in these time windows were computed. The performer depressed one of a series of footswitches to indicate when new content to the virtual environment should be added or removed. When new content was to be added, appropriate values of the statistics extracted from the performer's playing were rescaled to be parameter values transmitted to the selected algorithm to generate world content. For example, if the performer pressed a switch to request a new chamber, then the timing of the performer's playing in the longest time window (500 notes) was used to parameterise the algorithm which generated chambers. A chamber is a basic cube in whose walls are protuberances , the size and regularity of which are given by parameter values sent to the chamber generation algorithm. The more irregular and variable has been the playing of the performer in the 500 note time window, the more the protuberances break up the basic cubic shape to the chamber, and so forth. The general intention was to make the mean tendencies and variability in the performer's playing lead to graphical forms with comparable tendencies and variability. To give another example, highly syncopated, irregular playing would tend to lead to a variable, 'bendy' path for the viewpoint through the environment. Performing *Lightwork,* then, involved creating virtual environment content in real-time through an analysis of musical gesture (for more details, see Deliverable D2.2 from Year 1, and Bowers et al., 1998).

Working with *Lightwork* revealed a number of problems that we tried to address with some features of our design of *Blink. Lightwork* was realised with a back projection behind the performers. This showed the view from a single mobile viewpoint in the virtual environment. Although the viewpoint's motion was updated in response to performer gesture, it was quite possible for the image to 'run aground' in a part of the algorithmically generated virtual environment that didn't happen to be very interesting. Quite commonly the viewpoint might find itself close to or within a large scale virtual form which appeared as a series of large polygonal surfaces crashing through the screen. As there was no direct control over the projected image (only the algorithmically mediated control provided by the analysis of the performer's playing), performers often did not have adequate resources to avoid such circumstances.

In *Blink* we decided to learn from the experiments in inhabited TV in eRENA and work with a multi-camera paradigm. Rather than have the single mobile viewpoint of *Lightwork,* we experimented with three mobile virtual cameras in the *Blink* environments. In this way, we could always cut away from an unsatisfactory or boring image, or from a camera which was momentarily 'trapped' too close up against a large virtual form. (It should be noted that in neither piece have we worked with collision detection or algorithms designed to specifically avoid sub-optimal shots. There are three reasons for this. First, implementing collision detection would impose an excessive real-time penalty on the systems we are using, which would result in dramatically reduced frame-rates, especially with the complex image-rich forms we prefer to work with. Second, some of our virtual forms are generated by means of non-linear geometrical

algorithms that make it hard to determine bounding areas which collision detection could operate over; indeed, some involve taken a basic shape and turning it inside out multiple times, something which would lead to anomalous results for some implementations of collision detection. Finally, the visual effect of crashing through virtual forms to reveal whatever lies the other side is often exciting. That is, the very insensitivity which our path algorithms have to the exact positioning of virtual forms in the environment can lead to interesting shots just as often, indeed more often, than it leads to unappealing ones.) In *Blink* three different camera paths were defined and performers assumed something of a directorial role in viewing and previewing these and selecting one as the transmission (TX) image. As such *Blink* can be seen as offering a degree of real-time software editing in the transmission of image.

We also decided to learn from the experience of the puppetry workshop hosted by the International Puppetry Institute and the ZKM, reported in Chapter 1 of this deliverable. Chapter 1 makes a number of references (cf. Deliverable 2.3 from Year 1 of eRENA) to how problematic a large single screen can be for performers and audience to relate to, especially when it is unthinkingly inserted into a conventional stage space. Certainly, the richness of the visual experience in *Lightwork* was limited by the projection of just one viewpoint to the rear of the performers. In *Blink* we distributed no fewer than 21 large monitors around the environment of The Bomb, a somewhat cavernous nightclub in Nottingham which formed the venue for the event. In addition a large screen was placed in the main area of The Bomb where the performers of *Blink* were to be found along with 4hero and other technicians and DJs. The *Blink* software was run twice, on two quite separate systems, and an image was back projected to the large screen that was a video-mixed combination of the TX output from the two systems. In this way, we were able to fill the environment with a mass of image material, in multiple varying locations, at different levels of scale. The large screen image was an edited combination of two streams of image material that were themselves already edited. This routing of multiple image sources to multiple image destinations we hold to be a characteristic of electronic arenas.

In Deliverable D2.2 from Year 1, we note that it was a difficult matter to adequately calibrate the algorithms of *Lightwork* to make them responsive to performer gesture in a way that was appropriately visible to both performer and audience. Learning from this experience accordingly, in *Blink,* we worked with much simpler analyses of musical gesture to derive parameter values for the world construction algorithms. MIDI note data was simply analysed for its density in three different moving time windows (rather than separately analysing for pitch, velocity and timing). These values were normalised, rescaled and made available as parameter values for the world construction algorithms. In this way, we hoped to make features related to the density and variability of MIDI note events (e.g. tempo and density of rhythm) visible in terms of the density and regularity of visual forms in the virtual environments *Blink* generated (though, as we shall see, our intentions here were somewhat thwarted).

6.2. World Construction and Camera Control Interfaces

Figure 6.1 shows three screen shots of the *Blink* software in action, showing the world content and camera control window together with the preview view and the transmission (TX) view. The video out from the machine was configured to excerpt the middle portion of the TX view. (It should be noted we used very high resolution large Silicon Graphics monitors for *Blink*. For this reason, a whole screen shot suffers somewhat when reduced to the scale of this page!)

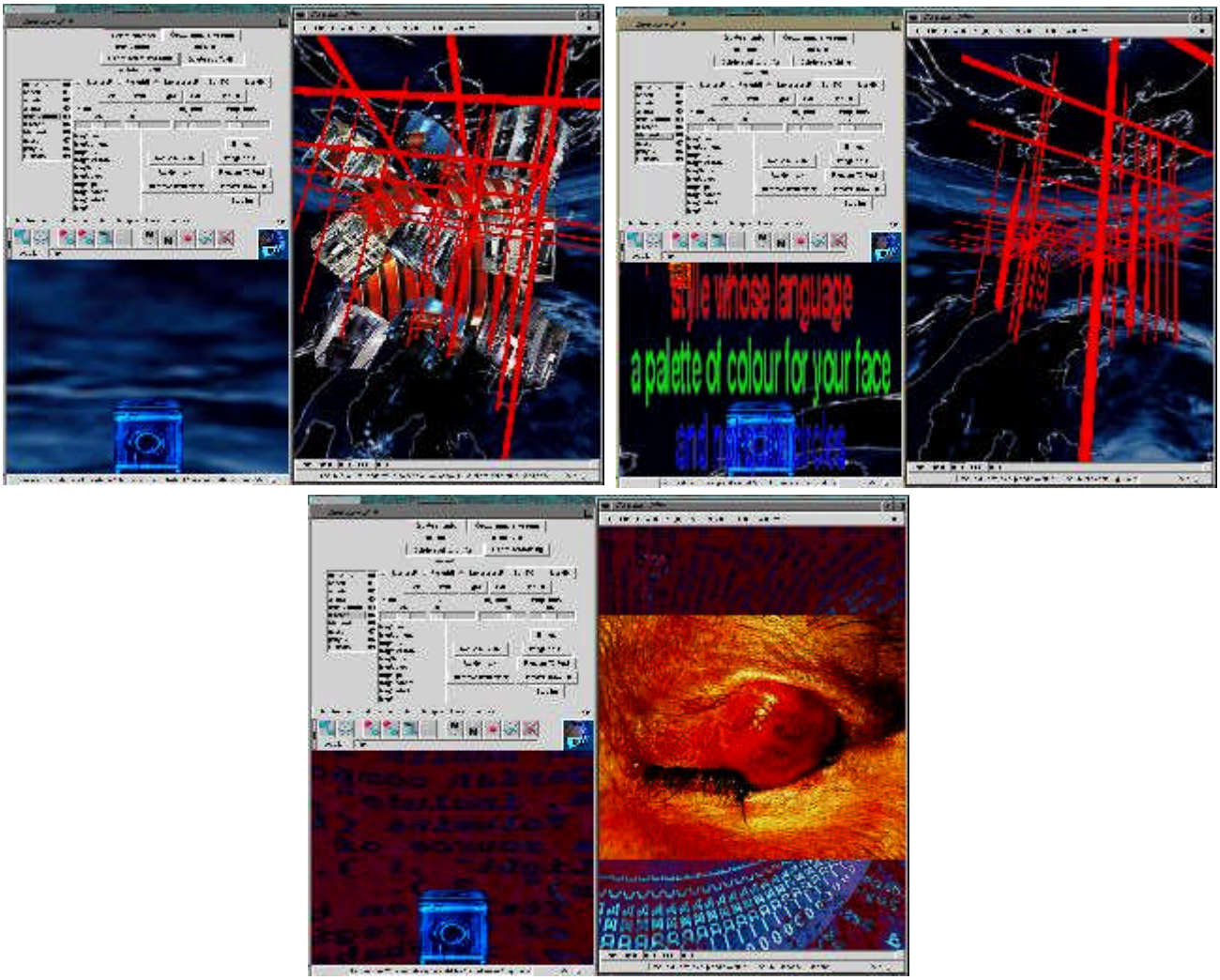

Figure 6.1: Three full screen shots of the Blink *interface. Content generation and camera control widgets are in the window to the top left of each screen. Below this is the preview window for the most recently selected camera. To the right is the transmission (TX) image. Note how in each case a view from a different camera than TX is being previewed.*

Figure 6.2 shows a closer, clearer image of the content generation and camera controls. Let us describe the operation of the *Blink* software by walking through these controls.

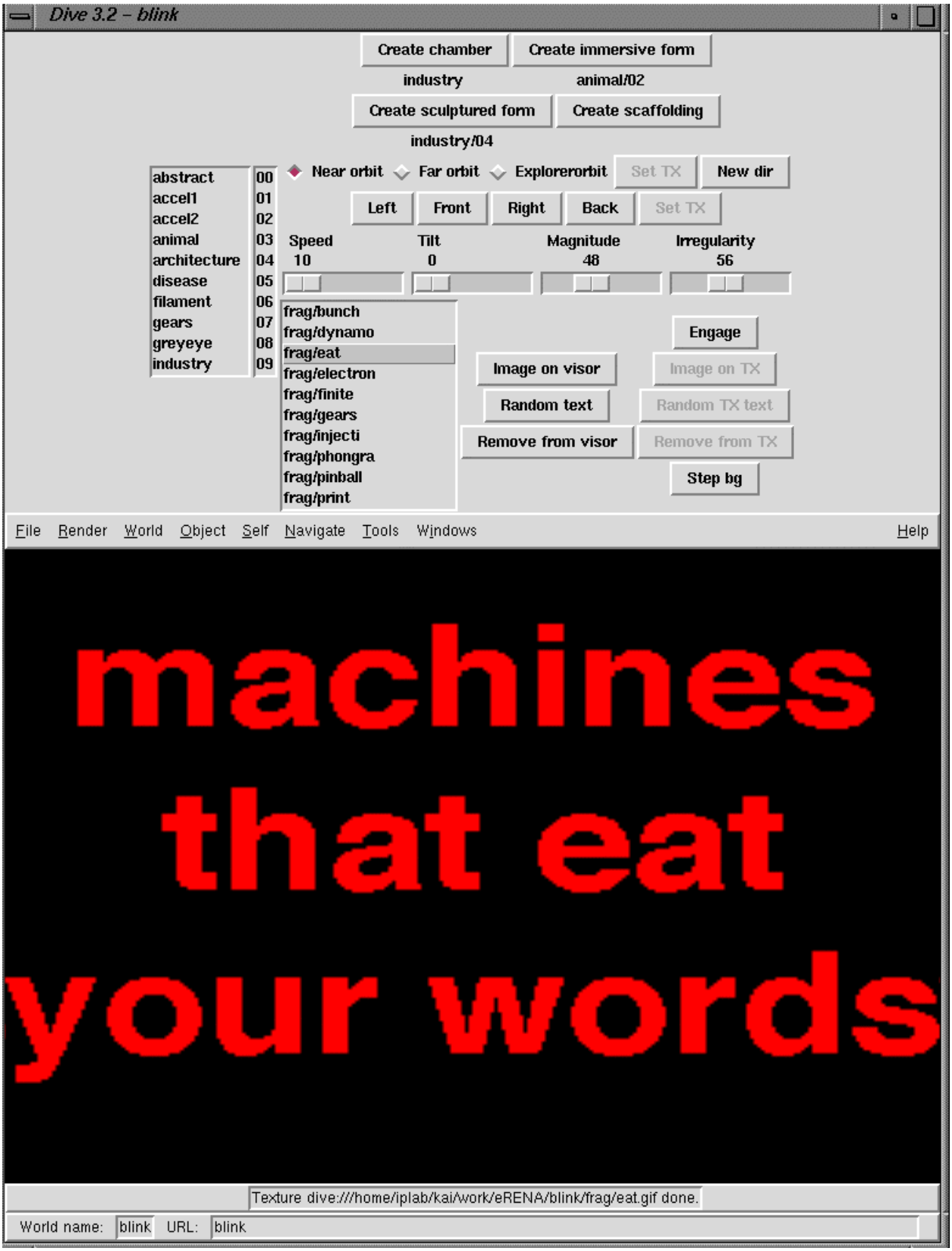

Figure 6.2: World contruction, camera control and preview window from Blink.

At the top of the window shown in Figure 6.2 are four buttons labelled ‘create chamber’, ‘create immersive form’, ‘create sculptured form’ and ‘create scaffolding’. When these buttons are pressed a chamber, immersive or sculptured form, or scaffolding is created. The labels on the buttons then ‘toggle’ to show ‘delete chamber’ and so forth, and another pressing will remove the item in question. Deliverable D2.2 describes these algorithms in more depth. For present purposes we refer the reader to the Gallery (Section 6.3) to see examples of the kind of content each algorithm can generate. Note the two fields (which are in fact scrollable) in the interface in Figure 6.2 which contain lists of names (one with ‘abstract’ at the top, one with ‘frag/bunch’). These refer to images which can be displayed within *Blink* in a number of ways. The list beginning with ‘abstract’ is a list of image archives, the images in which can appear as textures on objects in the virtual environment. Thus, it is possible to create a chamber with images as its ‘walls’. A selected image from an archive can also be mapped onto the sculptured and immersive forms. The image selection is made by using the number fields to the right of the archive names. The name of the last selected image or image archive appears under the button corresponding to the algorithm which used it. Thus a chamber containing images from the ‘industry’ archive was the last to be created at the time of the Figure 6.2 screen shot. We shall discuss the scrollable field with entries in it like ‘frag/bunch’ shortly.

Underneath the buttons for operating the content generation algorithms appears a choice between three different ‘orbits’. These define the path taken by a viewpoint through the virtual environment. ‘Far orbit’ and ‘near orbit’ define circular orbits far and near respectively from the centre of the virtual environment. ‘Explore orbit’ defines a path which leaves the centre in a random direction, heads towards the periphery of the virtual environment, then turns around and heads back towards the centre, where a new choice of random direction is made. (In contrast with *Lightwork*, we did not subject the camera paths to any performance-influenced change in notional radius, extent or modulation/‘bendiness’ in an attempt to simplify camera movements and lessen the occurrence of the ‘degenerate’ views we have already complained about.) When one of these buttons is depressed to select a camera path, the view from the camera is shown in the preview view below the controls. Changing the path makes no difference to the transmission (TX) view until the button next to the path selection alternatives labelled ‘Set TX’ is pressed. Only then does the TX view come to have the view controlled by the same path algorithm as the preview view. However, now there is freedom to experiment with new paths, before setting TX again. And so forth. The control ‘New dir’ changes the direction of rotation of the circular orbits (e.g. from clockwise to anti-clockwise) or causes the explore orbit to turn around and reverse its course. Such changes effect TX (and preview, if preview is engaged to the same orbit as TX) instantly. The controls ‘Left’, ‘Right’, ‘Front’ and ‘Back’ change the orientation of the view of the camera. Pressing ‘Back’ for example will show the virtual environment recede from the viewpoint. Again, this orientation can be experimented with independently of affecting TX. ‘Set TX’, alongside the orientation controls, has to be depressed for the TX to change.

The speed slider affects the speed with which the cameras move through the environment varying from very slow and sedate to a giddying pace. The tilt slider causes the plane of the camera’s movement to tilt between +/-90 degrees. The magnitude and irregularity sliders enable the magnitude and irregularity parameters for content generation to be set manually. It was intended that, in normal operation, *Blink* would take these values from analyses of the density of musical data. In the event of MIDI communications breaking down or unappealing results obtaining from the analysis, this manual override (which can be engaged by pressing ‘Engage’) was provided (this turned out to be vital).

The control labelled ‘Image on visor’ allows the display of an image selected in the other scrollable field to be displayed as if affixed to the ‘visor’ or ‘over the lens’ of the preview camera. An image placed here will occlude all that’s behind it (unless it contains transparent components itself). This enabled us to cut from mobile images taken from the virtual environment to inserted text, short phrases and other images. In Figure 6.2 we see an image with the text phrase ‘machines that eat your words’ placed ‘on the visor’. (‘Visor’ is actually some terminology from the DIVE VR system, which we use in our work, to designate a location where a proximal image can be displayed in this fashion.)

Pressing ‘Random text’ will generate a three-line ‘haiku’ style short text by making random selections between a set of possible first lines, a set of possible middle lines, and a set of possible last lines. The lines were all written by Mark Jarman, a poet who has collaborated with us on *Blink* and *Lightwork*. ‘Remove from visor’ will remove any text or other image positioned on the visor. As before, these images can be experimented with on the preview view without being displayed on TX. This allows the images to be browsed or various random three line texts to be generated before one is found which is usable. ‘Image on TX’ and ‘Random TX text’ will place the images and texts on the preview view onto TX while ‘Remove from TX’ restores the unobstructed view onto the virtual environment.

‘Step bg’ changes the image which serves as background to the whole virtual environment. Eight images are available in a circular queue. Finally, the menus in a row just above the preview window are the standard visualiser menus from DIVE, enabling us to call upon standard DIVE functionality if we needed to (e.g. to operate a 6DOF navigation controller).

6.3. *Blink* Image Gallery

Figures 6.3 to 6.10 show images taken from *Blink.*

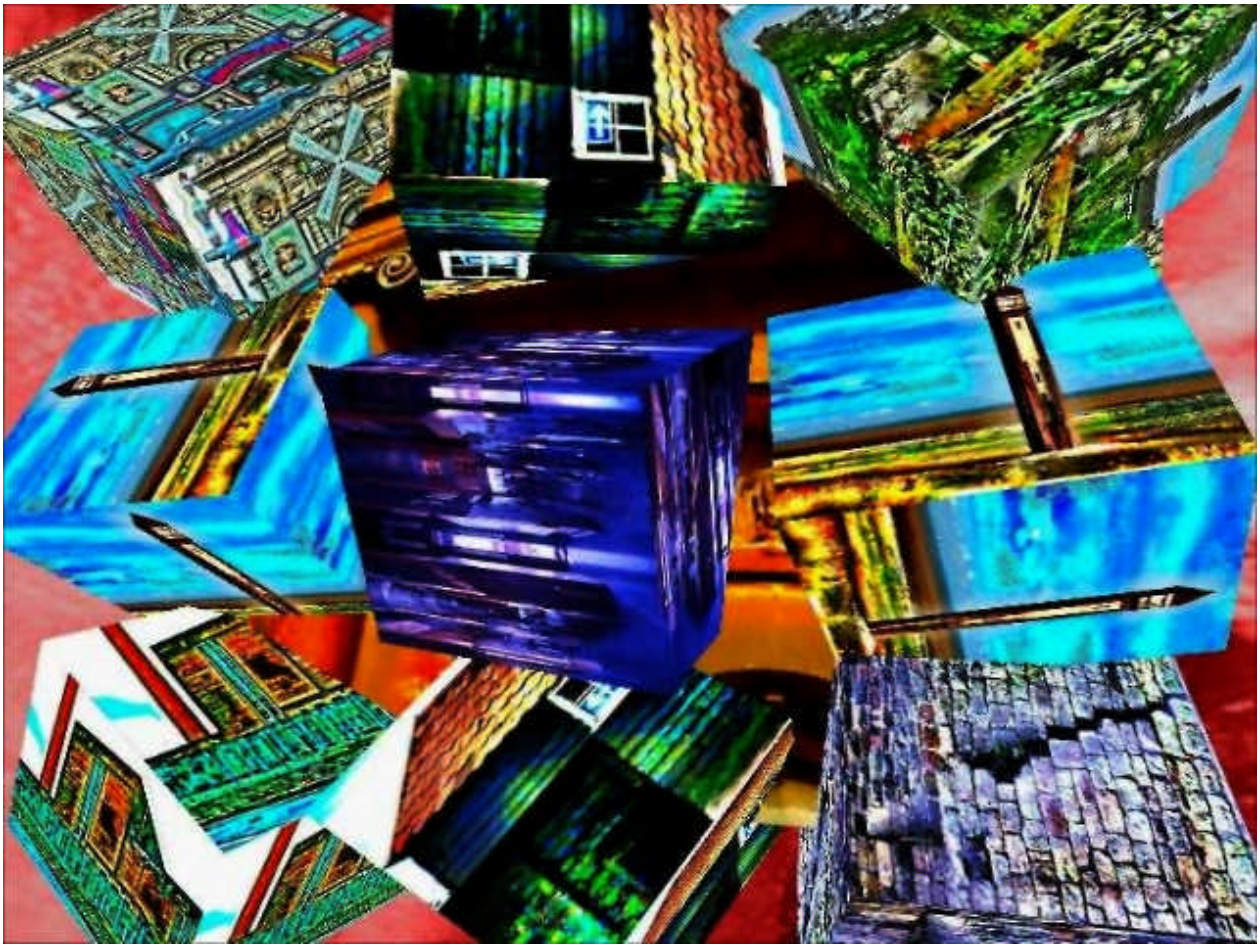

Figure 6.3: The external view (from the outer orbit looking inwards) of a chamber surfaced with images from the architecture image archive.

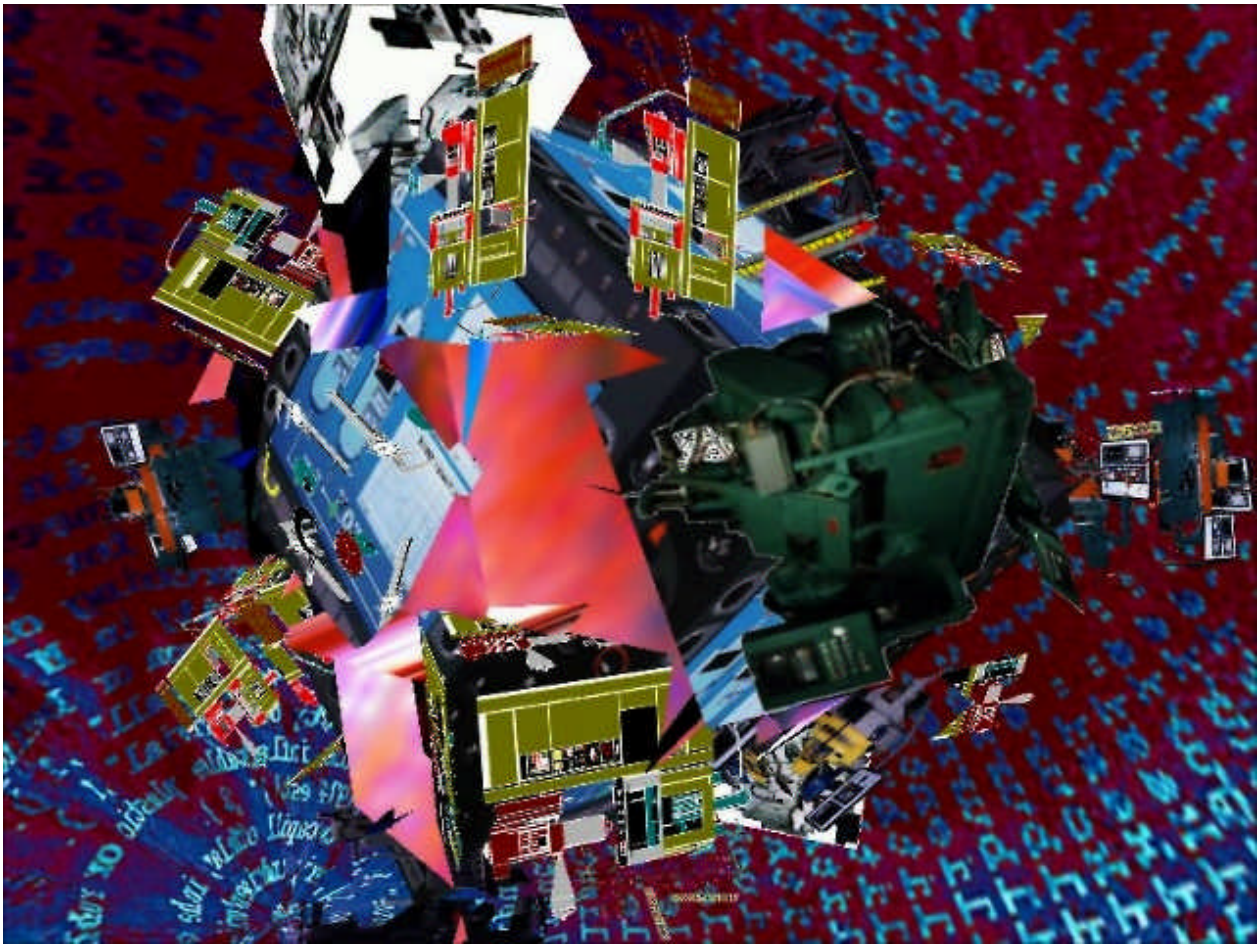

Figure 6.4: An external view of a chamber constructed using industry images, some of which contain some transparency. Note how the plane of the orbit has been tilted compared with Figure 6.3.

Figure 6.5: A sculptured form 'impaled' on scaffolding, near orbit. Sculptured forms use a graphical analogy of the frequency modulation equations of sound synthesis and radio to modulate a basic sphere (see Bowers et al., 1998, for more details).

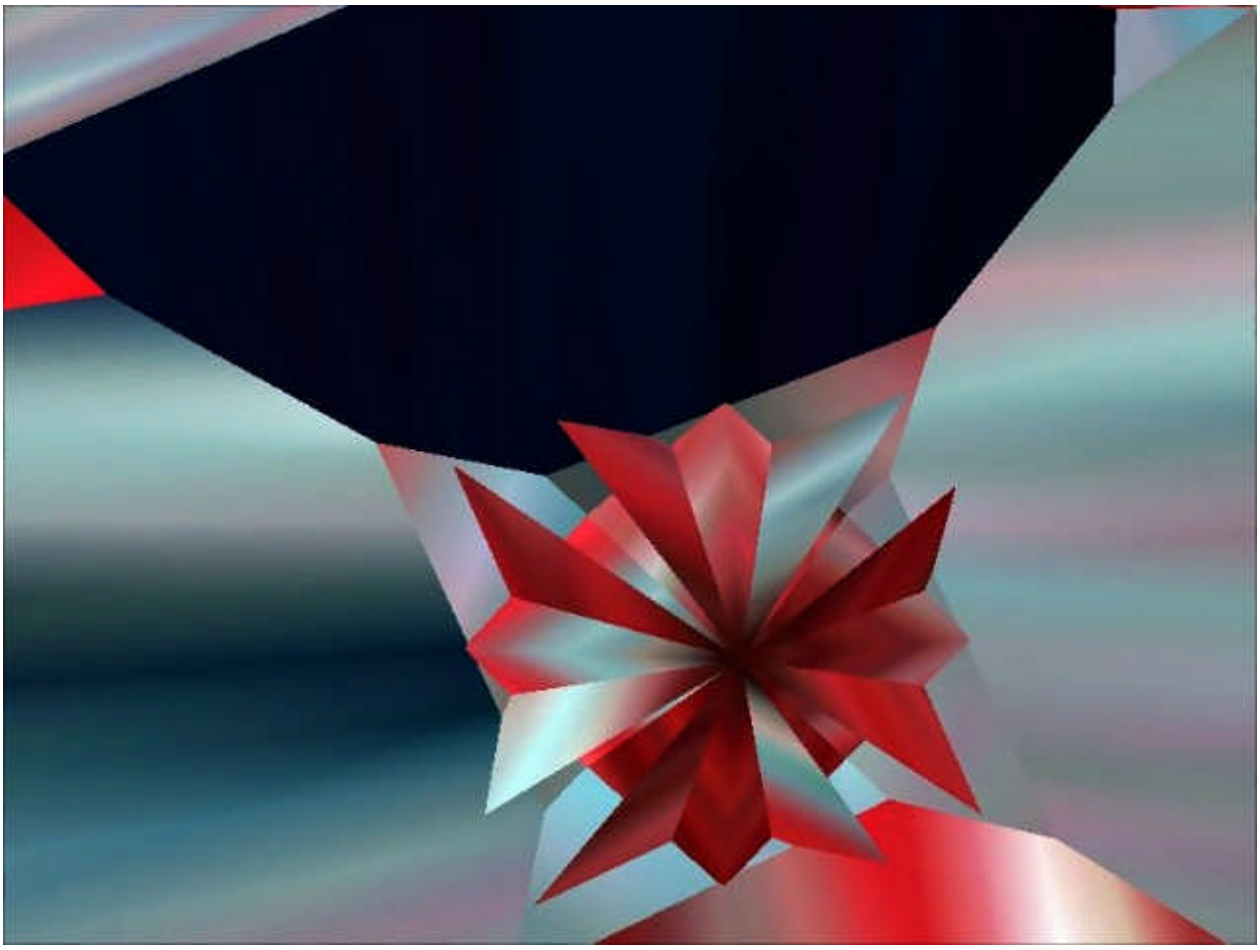

Figure 6.6: A very large immersive form. Here the viewpoint is surrounded by larger graphical surfaces. Immersive forms are created by using the same FM technique for form generation as sculptured forms (see Figure 6.5) but scaled to a higher range for the size of the form. View from an explorer orbit path.

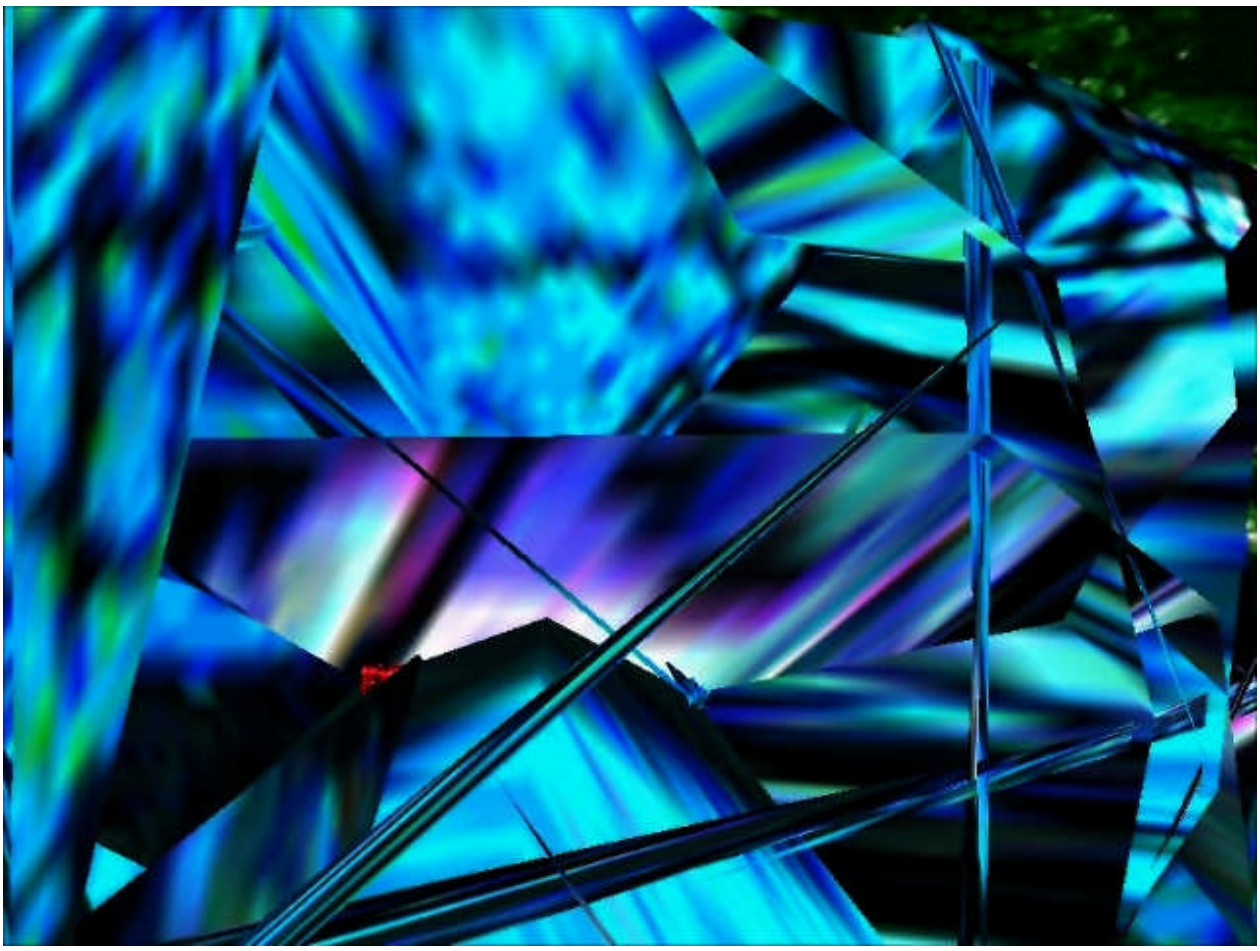

Figure 6.7: Immersive form viewed from an outer orbit.

Figure 6.8: Sculptured form, outer orbit.

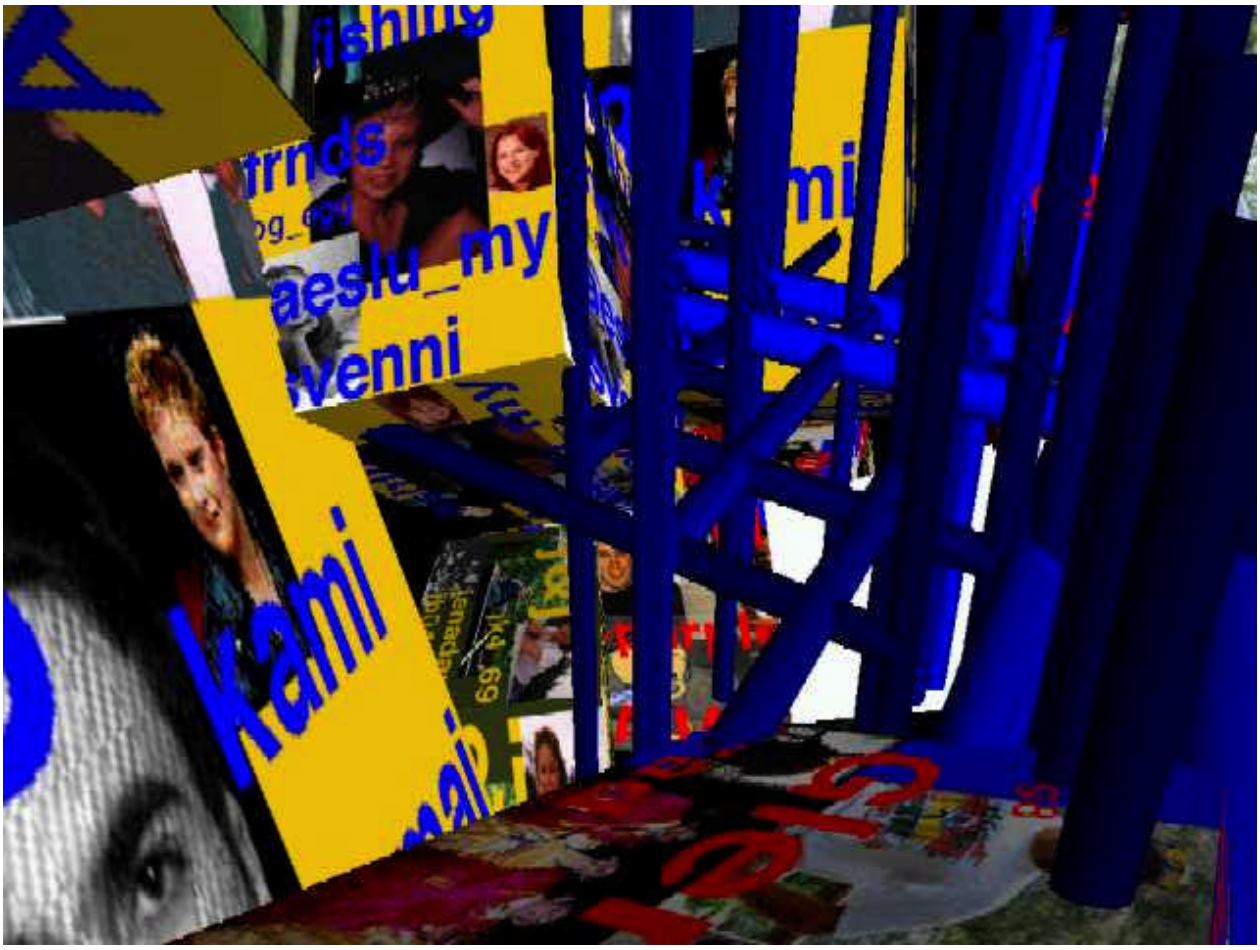

Figure 6.9: Using the explorer orbit, entering a chamber containing scaffolding.

Figure 6.10: Inside a chamber, inner orbit, looking inwards. The blue camera is a representation of the explorer camera as it passes through (each camera is distinctively coloured and visible in the other *cameras' views).*

6.4. Implementation, Installation and Performance Details

The *Blink* software is implemented in the DIVE VR system (http://www.sics.se/dive/) as a set of C programs and tcl scripts. For the Now98 performance two Silicon Graphics O2s were used, each running a separate software installation. The multi-camera concept in *Blink* was implemented in DIVE by having two visualisers run on each machine, one to render TX, one to render the preview view, the notional location and orientation of the other cameras being calculated but not rendered. John Bowers and Mike Craven performed *Blink* by each interacting with one of the O2s, generating world content, controlling cameras and selecting views. Video was taken from each machine and mixed, using conventional video mixing technology, to a large back projection screen by Jim Purbrick.

4hero, a group of drum and bass musicians, performed live in The Bomb with Carl Craig appearing 'by satellite' from his studio in Detroit. Derek Richards managed the intercontinental links, oversaw local networking and served as something of a stage manager during the event. It was intended that 4hero and Carl Craig would exchange MIDI information so as to interact with each other's musical equipment while playing together. Ultimately, this had to be abandoned as MIDI communications were not reliable enough and some of the MIDI data output by 4hero's equipment seemed corrupted. The merged MIDI data from both sets of performers was to be input to the MIDI analysis program we had authored running in the Opcode MAX music and multimedia programming language and residing on two Macinstosh computers, one corresponding to each *Blink* installation. The analysis program was a development and a simplification of the 'Interactive Narrative Machine' program used in *Lightwork.* Unfortunately, as MIDI communications proved unreliable, our concepts for the parameterisation of graphical algorithms by live music were not properly tested. Accordingly, we do not go into further detail here about the software developed for MIDI analysis.

Twenty one monitors were arranged around the public spaces in The Bomb. Groups of three monitors were sited in seven locations such that a visitor to the club would have line of sight to at least one monitor wherever they were and from most locations a group of three could be seen. One of each group showed the live image of Carl Craig in Detroit, the other two showed the worlds from the two independent *Blink* installations.

To structure the performance John Bowers and Mike Craven agreed a running order of some ten episodes during a performance which was anticipated to last for one hour. (Before and after the performance, the software was still worked with but set to show more 2D content and with a slower rate of change of material.) The performance was structured on a principle of gradually revealing the kinds of virtual environments which could be generated and steadily increasing their complexity. We tried to ensure that both installations would be showing material from different algorithms and image sources at any one moment, though both would be approximately co-ordinated in the 'tempo' of camera movement, cutting and world change so as to steadily increase the visual tension and dynamism as the hour progressed. Within this general framework, the performance was to be improvised. For his part, Jim Purbrick, editing the two streams from the two *Blink* installations, was to edit from one source to the other as he felt fit, again increasing the pace of editing as the performance continued.

6.5. Experience in Performance

The event was a very popular inclusion in the Now98 arts festival. It was sold out several days in advance and a very enthusiastic audience greeted 4hero and Carl Craig. There were numerous difficulties co-ordinating across the Atlantic right down to the start of the performance itself but this didn't dampen the enthusiasm of the audience. If anything the rather hesitant start to such a high-tech event added to its charm. Overall the audience reception to the event was extremely enthusiastic but it is hard to know what of this was due to the musicians, both acts being internationally known and revered in their fields, and what our performance of *Blink* added to this. The club setting militated against conducting collective post-event discussions with the audience as, for example, has been done in eRENA with *Out Of This World* and *Murmuring Fields* (see Deliverables D7a.1 and D6.2 respectively). What we could do was to ask some arbitrarily selected audience members specifically about the VR work we had done and seek out their appraisals.

For the most part, reactions were very favourable. However, some critical comments did appear concerning the pace of the editing which several people found rather slow to be idiomatic for the kind of music being played by 4hero and Carl Craig (fast drum and bass and techno). Certainly, we were cutting *Blink* much slower than the average video accompaniment to music of this sort. As such, our image-work did not quite have the excitement that some audience members might have expected. Generally, though, our efforts to create a multi-screen environment with 'transmissions' all around the venue was very well received. It clearly gave the venue an identity and an indication that something different was occurring. Most people questioned also found the content of the image material and the fact that it was clearly sourced from a virtual environment unusual and stimulating.

From a performer's viewpoint, the remarks about the pace of editing make sense. It turned out to be rather cumbersome to manipulate the on-screen widget based interfaces of *Blink*. We would have liked to have been able to cut with greater rapidity, both for aesthetic effect and to get out of trouble when cameras were showing unappealing input. As the interface was hard to manipulate, it was also hard to work up noticeable changes in editing pace for dramatic effect.

The difficulty of working with the interface is probably due to a number of features. Interacting with on-screen buttons was all mouse-driven. Manipulating the mouse, then, became the bottleneck for performer gesture and much of the time was just too slow. It was also error prone, being easy to over or under-reach a mouse gesture and press the next-door button. As such the slowness of mouse-gestures contrasted greatly with the facility with which the conventional video mixer could be handled. Jim Purbrick was able to vary pace in the editing of content for the large screen in ways which sometimes compensated for the relative slowness of the changes in image point of view coming from the VR systems.

Some of our tasks at the interface also required a number of operations. For example, we might have to select an image, test it in preview, then transmit it. This was a cumbersome cycle. Anticipating this in advance to some degree, we decided to make some operation directly effect TX, while others required 'setting up', and yet others could be done either way. While this speeded some operations, it introduced a degree of inconsistency in how we had mapped tasks to interface operations, again on occasion leading to errors.

Mousing around the screen also required a degree of visual attentiveness, especially once we became aware that we were making the occasional error. But this distratcted from visually

inspecting the image, to ensure that what was 'going out' was what we wanted. Generally, we would suggest that a mouse-driven interface is inappropriate for the kind of task we are presenting users/performers with here.

In a technical sense, our applications worked reliably enough. However, we did experience some anomalies in the behaviours of the virtual environments, which we believe we have inherited from DIVE and/or the visualiser to the system we were using (the standard release visualiser, 'vishnu'). For example, if we placed an image with some transparency on the visualiser, this image occluded all behind it apart from the background image. Clearly, this behaviour reflects the rendering order in vishnu. This did not amount to a major problem in performance. More significant was the lack of synchronisation between visualisers which we experienced. If we set TX to show the view from the preview camera, the views were very unlikely to be identical. One would be leading the other to some degree. This had the consequence that our edits could not be as precise as we would have wished and the transitions between cameras was made more on the basis of general changes between their views than the precise dramatic effect it would be possible to achieve with tighter coupling.

This is unfortunate because both of the performers/users of the *Blink* software had worked out *sequences* of shots which would work effectively. For example, a cut from the explorer camera to a inner orbit camera made just as the explorer is entering through the chamber walls gives the impression of a smooth edit (a kind of 'match on crash' to adapt film terminology!). This was possible to achieve in general terms but precise matches of visual content were not possible.

Let us now turn to some more general issues of interest to eRENA and worthy of review in *Blink.* Although the performance was largely improvisatory, we did work with a loose running order. Some of the work done by this running order could have been done technically. That is, we could have had some technical support in managing the ten or so episodes we intended in our performance. In short, the *Blink* software could have benefited from some 'event management' support much like that developed for *OOTW*. For how we structured *Blink* on this occasion, event management by listing phases would have sufficed, though this would not be only the possibility. For example, just as we intended the gestures of musical performers (at least as revealed in analysis of their MIDI data) to parameterise the algorithms, perhaps similar sources could influence the unfolding of the 'narrative' to an event. We explored ideas similar to this in *Lightwork* under the banner of 'narrative from within', the idea being that one's interaction with a system is both the realisation and creation of a narrative structure.

Several developments in eRENA have been concerned with the relationship between media, say, sound and vision. Failures of MIDI communications at this event inhibited our explorations of directly driving *Blink* with musical data. Accordingly, we had to manually parameterise our algorithms. This added to our manual burden as performers, especially as it is not easy to translate changes in musical features to abstract parameters of non-linear graphical algorithms! However, in a sense this adds to the remarks we made in our evaluation of *Lightwork* (Deliverable D2.2) about the importance of combining direct methods of interaction with algorithmically mediated methods. The *gestural link* between the on-screen manipulation of parameter values and graphical content was not clear enough to us as performers for that link to mediate a further link between music and image intended to be present for the audience. For a variety of reasons, now, we feel that it is important to combine direct gestural links (touch) with more indirect forms of interaction (shaping parameters and the behaviour of algorithms).

6.6. Conclusions and Future Work

Our conclusions are reminiscent of those made by partners involved in inhabited TV experiments in review of their early work at the end of Year1 of eRENA. We believe we have been successful in creating, in the *Blink* event, an environment rich in sound and image, with image material distributed throughout a physical environment. We believe we have conducted an interesting experiment in bringing *montage* (or at least cutting!) to VR and in working with a multi-camera paradigm in an artistically oriented VR performance. We believe also we have demonstrated the potential for interactively constructing some aspects of the content of a performance in an electronic arena in real-time.

In short the feasibility of what we are attempting has been shown. And yet some critical details which make for satisfactory events and experiences of them are not fully attended to. Many of these concern the pacing of an event and its controlled variation. We have suggested enhancements and changes to our work to deal with these issues, e.g. by incorporating some event management mechanism. Much of our work in Year 3 of eRENA will concern thinking about event management mechanisms for events of the sort we have demonstrated here: artistically oriented improvisatory events. It is important to also consider event management requirements from this perspective to ensure that the notations and techniques demonstrated so far in eRENA (see Deliverable D7a.1) do not prove to be overly restrictive. For this reason, the active experimentation with systems for choreographing user input under development at ZKM (see Chapter 2 of this deliverable) is of extreme interest. Working up the structure of an event through the choreography of user input seems an interesting alternative orientation to the imposition of structure through top-down controls realised as state-transitions.

While event management support would help with questions of pace, so would the more careful design of alternative interfaces. We have seen that mouse-driven interfaces are problematic when one is performing a combined role of camera operators, director and world creator (as one is in *Blink*). This is why the work reported in Chapter 5 of this deliverable is of importance as it opens out possibilities for more appropriate physical interfaces for applications like this.

In summary, *Blink* enabled us to demonstrate the feasibility of a multi-camera approach to an artistically oriented experiment in virtual reality where those mutiple cameras and editing between them is accomplished in software. We believe we created a rich augmented physical environment which is in many ways prescient of what some kinds of electronic arena will be. We showed how visual content can be generated for such environments from source materials and algorithms which assemble them. We hope we have shown the variety of forms that it is possible to create thereby and have demonstrated the interest in studying technological support for settings where the creation of content is in great part a real-time affair, i.e. improvisatory settings. In creating this event, we have learned greatly from the experience of others in the eRENA project as well as building on and responding to our own experience. We find ourselves in line with an agenda common to many efforts in eRENA which focuses on questions of pace and engagement, and seeks to supply an appropriate degree of technical support to facilitate effective real-time solutions.

6.7. Acknowledgments

The authors thank Derek Richards and Andrew Chetty for their organisation of the *Digital Clubbing* event at Now98 which *Blink* formed a part of. We also thank our eRENA partners at Nottingham for the loan of machines for the event and providing us with facilities while we refined our software in the lead-up to the event. We are also grateful for the invaluable help of Mike Craven and Jim Purbrick in assisting the first author in performing *Blink*.

References

[eRENA3.1] Demonstration and Evaluation of Inhabited Television, eRENA Deliverable 3.1, 1998.

[HREF1] "Questions and Answers about…Audience Participation" http://wwwphish.net/PhishFAQ/laudience.html

[HREF2] Piro, S., "It was great when it began", http://rockyhorror.com/partbegan.html

[HREF3] "Audience Participation creates enchanting atmosphere at opera", http://www.cc.govt.nz/MediaReleases/98audience.html

[HREF4] "Option Finder: Product Information", http://www.optionfinder.com/product.line/product.content.html

[HREF5] "CINEMATRIX: Interactive Entertainment Systems", http://www.cinematrix.com/

[HREF6] "Ars Electronica: Loren & Rachel Carpenter Audience Participation", http://www.aec.at/fest/fest94e/carp.html

Ackerman, Mark S., editor (1996). *CSCW '96: Cooperating Communities*, November 1996. http://www.acm.org/pubs/contents/proceedings/cscw/240080/index.html.

Alias | Wavefront (1998). "Using Maya: Animation Version 1.0" and "Using Maya: Dynamics Version 1.0".

Bandi, Srikanth and Thalmann, Daniel (1995). "An adaptive spatial subdivision of the object space for fast collision detection of animated rigid bodies". *Computer Graphics Forum*, 14(3): C-259–C-270, 1995. Proceedings Eurographics '95.

Barrus, J. and Anderson, D. (1996). "Locales and Beacons." in *Proc. 1996 IEEE Virtual Reality Annual International Symposium (VRAIS'96)*, San Jose, 1996.

Benford, Steve, Bowers, John, Fahlén, Lennart E., and Rodden, Tom (1994). "A spatial model of awareness". In Björn Pehrson and Eva Skarbäck, editors, *Proceedings of 6th ERCIM Workshops.* ERCIM, June 1994.

Benford, Steve, Bowers, John, Craven, Mike, Greenhalgh, Chris, Morphett, Jason, Regan, Tim, Walker, Graham, and Wyver, John (1999). *Evaluating Out Of This World: An Experiment in Inhabited Television.* eRENA, February 1999. Deliverable 7a.1.

Benford, Steve, Brazier, Claire-Janine, Brown, Chris, Craven, Michael, Greenhalgh, Chris, Morphett, Jason, and Wyver, John (1998). *Demonstration and Evaluation of Inhabited Television.* eRENA, May 1998. Deliverable 3.1.

Benford, S., D. and Fahlén L. E. (1993). "A Spatial Model of Interaction in Large Scale Virtual Environments", *Proc. Third European Conference On Computer Supported Co-operative Working (ECSCW '93)*, Milano, Italy, Kluwer Academic Publishers, pp. 109–124.

Benford S., Greenhalgh C., Brown C., Walker G., Regan T., Morphett J., Wyver J. and Rea P. (1998). *Experiments in inhabited TV*, Proceedings of CHI'98, pp 289–290.

Benford, S. D., Greenhalgh, C. M. and Lloyd, D. (1997). "Crowded Collaborative Environments", *Proc. CHI'97*, Atlanta, pp. 59–66, USA, March 1997, ACM Press.

Benford, S. D., Greenhalgh, C. M., Snowdon, D. N, and Bullock, A. N. (1997). “Staging a Poetry Performance in a Collaborative Virtual Environment”, *Proc. ECSCW'97*, Lancaster, UK, Kluwer. 1997.

Benford, S., Greenhalgh, C., Craven, M., Walker, G., Regan, T., Morphett, J., Wyver, J. and Bowers, J. (1999). “Broadcasting on-line social interaction as inhabited television”. Submitted to ECSCW99.

Bilson, A. J. (1995). “Get into the Groove: Designing for Participation”, *Interactions*, April 1995.

Blinn, J. (1988). “Where am I? What am I looking at?”. *IEEE Computer Graphics and Applications*, July 1988, pp. 76–81.

Bly, S., Harrison, S. and Irwin, S. (1993). “Media Spaces: Bringing People Together in a Video, Audio and Computing Environment”, *Communications of the ACM*, Vol 36, Number 1, pp. 28–47, 1993.

Boulic, R., Capin, T., Huang, Z., Moccozet, L., Molet, T., Kalra, B. Lintermann, B., Magnenat-Thalmann, N., Pandzic, I., Saar, K., Schmitt, A., Shen, J. and Thalmann, D. (1995). “The HUMANOID Environment for Interactive Animation of Multiple Deformable Human Characters”, *Proc. Eurographics*, Maastricht, 1995.

Bowers, J. (1994). “The work to make the network work”. *Proc. CSCW'94*, 1994.

Bowers, J., O'Brien, J. and Pycock, J. (1996). “Practically accomplishing immersion”. *Proc. CSCW'96*, 1996.

Bowers, John, Pycock, James, and O'Brien, Jon (1996). “Talk and embodiment in collaborative virtual environments”. In *CHI '96: Human Factors in Computing Systems: Common Ground*, pages 58–65. SIGCHI, April 1996. http://www.acm.org/pubs/articles/proceedings/chi/238386/p58-bowers/p58-bowers.html.

Bregman, A. S. (1990). *Auditory scene analysis*. Cambridge, MA: MIT Press.

Capin, T.K., Pandzic, I.S., Noser, H., Magnenat Thalmann, N. and Thalmann, D. (1997). “Virtual Human Representation and Communication in VLNet Networked Virtual Environments”, *IEEE Computer Graphics and Applications*, 1997.

Chalmers, M. (1994). “Information environments”. In L. MacDonald & J. Vince (eds), *Interacting with virtual environments*, Chichester, 1994.

Chen, M., Mountford, J. and Sellen, A. (1988). “A Study in Interactive 3-D Rotation Using 2-D Control Devices”, *Computer Graphics*, 22(4), August 1988, pp. 121–129.

Concise Oxford Dictionary (1996). , Oxford University Press, ISBN: CN-9129, 1996.

Damer, B. (1997). “Demonstration and Guided Tours of Virtual Worlds on the Internet”, CHI’97 (demonstrations), Atlanta, US, 1997.

Deussen O., Hanrahan P., Lintermann B., Mech R., Pharr M. and Prusinkiewicz P. (1998). “Realistic Modeling and Rendering of Plant Ecosystems”, *Proc. SIGGRAPH '98*, Orlando, Florida, 1998.

Deussen, O. and Lintermann, B. (1997). “A Modelling Method and Interface for Creating Plants”, *Proc. Graphics Interface '97*, Kelowna B.C., Morgan Kaufmann Publishers, May 1997.

Deussen, O. and Lintermann, B. (1997) “Erzeugung komplexer botanischer Objekte in der Computergraphik”, in *Informatik Spektrum* 20/4, Berlin-Heidelberg-New York, Springer-Verlag, October 1997.

Deussen, O., Lintermann, B. and Prusinkiewicz, P. (forthcoming). “Computergenerierte Pflanzen” (working title), *Spektrum der Wissenschaft* (*Scientific American*, international issue in German). In preparation (publication scheduled summer 1999).

Drucker S. (1994), Intelligent Camera Control for Graphical Environments, PhD Thesis, MIT Media Lab. 1994.

Drucker, S., Galyean, T. and Zeltzer, D. (1992). “CINEMA: A System for Procedural Camera Movements”, in *Proc. 1992 Symposium on Interactive 3D Graphics*, Cambridge MA: ACM Press, March 29–April 1, 1992, pp. 67–70.

Drucker, Steven M. and Zeltzer, David (1995). “CamDroid: A system for implementing intelligent camera control”. *Computer Graphics*, 29:139–144, April 1995. http://www.acm.org/pubs/articles/proceedings/graph/199404/p139-drucker/p139-drucker.pdf.

Elsaesser, T. and Hoffman, K. (eds.) (1998). *Cinema Futures: Cain, Abel or Cable.* Amsterdam Univ. Press. 1998.

Fairchild, Kim Michael, Poston, Timothy, and Bricken, William (1994). “Efficient virtual collision detection for multiple users in large virtual spaces”. In Gurminder Singh, Steven K. Feiner, and Daniel Thalmann, editors, *Virtual Reality Software & Technology*, pages 271–285. ACM & ISS, World Scientific, 1994.

Feiner, S.K. and McKeown, K.R (1991). “Automating the generation of coordinated multimedia explanations”. *IEEE Computer* 24(10), 1991, pp 33–41.

Fishkin K., Moran T. and Harrison B. (1998). “Embodied User Interfaces: Towards Invisible User Interfaces”, *Proc. EHCI'98 Conference on Engineering for Human-Computer Interaction*, Crete Greece, 1998

Fitch, T., and Kramer, G. (1994). “Sonifying the Body Electric: Superiority of an Auditory over a Visual Display in a Complex, Multivariate System.” In Kramer (1994).

Fitzmaurice G., Ishii H. and Buxton W. (1995). “Bricks: Laying the Foundations for Graspable User Interfaces”, *ACM Proceedings CHI’95*, Denver, Colorado, 1995.

Flowers, J. H., Buhman, D. C., & Turnage, K. D. (1996). “Cross-modal equivalence of visual and auditory scatterplots exploring bivariate data samples”. *Human Factors*, 39(3), pp 341–351.

Frécon, E., Eriksson, H. and Carlsson, C. (1992). “Audio and Video Communication in Distributed Virtual Environments”, *Proceedings of the 5th MultiG Workshop*, December, 1992.

Funkhouser, Thomas A. (1996). “Network Topologies for Scalable Multi-User Virtual Environments”, *in Proc. 1996 IEEE Virtual Reality Annual International Symposium (VRAIS’96)*, San Jose, CA, April, 1996.

Gabor, D. (1947). “Acoustical Quanta and the Theory of Hearing.” *Nature*, Vol. 159, No. 4044.

Garfinkel, H. (1967). *Studies in ethnomethodology*. Prentice Hall. 1967.

Gaver, W. W. (1994). “Using and creating auditory icons.” In Kramer (1994), pp. 417–446.

Gaver, W. W., Smith, R. B., and O'Shea, T. (1991). "Effective sounds in complex systems: the ARKola simulation." In Proceedings of CHI '91. New York: ACM

Gleicher, M. and Witkin, A. (1992). "Through-the-Lens Camera Control", *Computer Graphics* (*Proc. Siggraph*), Vol. 26, No. 2, Aug. 1992. pp. 331–340.

Greenhalgh, C. and Benford, S. (1995). "MASSIVE: A Virtual Reality System for Tele-conferencing", *ACM TOCHI*, 2 (3), pp. 239–261, ACM Press, 1995.

Greenhalgh, C. and Benford, S. (1997). "A Multicast Network Architecture for Large Scale Collaborative Virtual Environments", *Multimedia Applications and Services and Techniques—ECMAST '97, Proc. 2nd European Conference*, Serge Fdida and Michele Morganti (eds.), Milan, Italy, May 21–23 1997, pp 113–128, Springer.

Greenhalgh C., Benford S., Taylor I., Bowers J. et. al. (1999). "Creating a Live Broadcast from a Virtual Environment", *SIGGRAPH '99, Conference Proceedings*, Los Angeles, 1999.

Han, J. and Smith, B. (1996). "CuSeeMeeVR Immersive Desktop Teleconferencing", *ACM Multimedia '96*, Boston, MA, USA, ACM 0-89791-871-1, pp.199–207, 1996.

Hayward, Chris (1994). "Listening to the Earth Sing." In Kramer (1994).

He, L., Cohen, M.F., Salesin, D.H. (1996) "The Virtual Cinematographer: A Paradigm for Automatic real-Time Camera Control and Directing". *Proceedings of SIGGRAPH 96*

Hoch M. (1997). "Object Oriented Design of the Intuitive Interface", , *Proceedings 3D Image Analysis and Synthesis*, Erlangen, November 17–19, Infix 1997, pp 161–167.

Hoch M. (1998). "A Prototype System for Intuitive Film Planning", *Third IEEE International Conference on Automatic Face and Gesture Recognition (FG'98)*, April 14–16, Nara, Japan 1998, pp 504–509.

Hoppe, Axel, Gatzky, Thomas, and Strothotte, Thomas (1995). "Computer supported staging of photorealistic animations". In *GraphiCon '95 Proceedings* (St. Petersburg, July 1995), volume 1, pages 179–185, St. Petersburg, July 1995. GRAFO Computer Graphics Society.

Hughes, J., Randall, D. and Shapiro, D. (1992). "Faltering from ethnography to design". *Proc. CSCW'92*, 1992.

Ishii H. and Ullmer B. (1997). "Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms", *ACM Proceedings of CHI'97*, pp. 234–241.

Karp, Peter and Feiner, Steven (1990). "Issues in the automated generation of animated presentations". *Proceedings of GI '90*. pp. 39–48.

Kozel, S. (1998). "Marionettes and dancers, dance and digital technologies", in *Archis* 61, Amsterdam, 1998/10

Kramer, G., editor (1994). Auditory Display: Sonification, Audification, and Auditory Interfaces. Addison-Wesley, Reading (MA), 1994.

Kramer, Gregory et al (1997). Sonification Report: Status of the Field and Research Agenda, 1997.

Lea, R., Honda, Y. and Matsuda, K. (1997). “Virtual Society: Collaboration in 3D Spaces on the Internet”, *Computer Supported Co-operative Work: The Journal of Collaborative Computing,* 6, 227–250, 1997, Kluwer.

Lintermann, B. and Deussen, O. (1996). “Interactive Modeling of Branching Structures”, *SIGGRAPH 96 Visual Proceedings*, New Orleans, 1996

Lintermann, B. and Deussen, O. (1996). “Interactive Modeling and Animation of Branching Botanical Structures”, EGCAS'96 Workshop on Computer Animation and Simulation, published in *Computer Animation and Simulation '96,* Vienna-New York, Springer, 1996, pages 139–151

Lintermann, B. (1997). *Morphogenesis*, interactive installation for the ZKM Medienmuseum; exhibited at *Multimediale 5*, Oct–Nov 1997. http://i31www.ira.uka.de/~linter/morphogenesis.html

Lintermann, B. and Deussen, O. (1998). “A Modeling Method and Interface for Creating Plants”, *Computer Graphics Forum*, vol 17, number 1, March 1998

Lintermann, B. and Belschner, T. (1998). *Sonomorphis*, interactive installation with genetic graphics and sound; exhibited at *surrogate1,* ZKM Institute for Visual Media, Nov–Dec 1998. http://www.zkm.de/surrogate/sonomorphis.html

Lintermann, B. and Deussen, O. (1999). “Interactive Structural and Geometrical Modeling of Plants”, *IEEE Computer Graphics & Applications*, vol 19(1), Jan/Feb 1999

Lunney, D. and Morrison, R. C. (1981). “High-Technology Laboratory Aids for Visually Handicapped Chemistry Students.” *J. Chem. Ed.* 58(3). pp 228–231.

Macedonia, M. R., Zyda, M. J., Pratt, D. R., Brutzman, D. P. and Barham, P. T. (1995). “Exploiting Reality with Multicast Groups: A Network Architecture for Large-scale Virtual Environments”, *Proc. VRAIS’95*, 11–15 March, 1995, RTP, North Carolina.

Mackinlay, J., Card, S. and Robertson, G. (1990). “Rapid Controlled Movement Through a Virtual 3D Workspace”, *Computer Graphics*, 24(4), August 1990, pp. 171–176.

Manovich, L. (1998). “Towards an archaeology of the computer screen & To lie and to act: Cinema and telepresence”. In Elsaesser and Hoffman (1998).

Martin D., Bowers J. and Wastell D. (1997). The interactional affordances of technology: an ethnography of human-computer interaction in an ambulance control centre. *Proceedings of HCI'97*.

Morphett, J. (1998). “eTV: A Mixed Reality Interface onto Inhabited TV”, ESPRIT project #25379 (eRENA) deliverable D3.2, 1998.

Mulder, J.D. and van Wijk, J.J. (1995). “3D Computational Steering with Parametrized Geometric Objects”. In Nielson, G.M. and Silver, D. (eds.), *Proceedings Visualization'95*, IEEE Computer Society Press, Los Alamitos, CA, 1995, pp 304–311.

Norman, S.J. (1989). “Les poupées pixel: marionnettes du troisième millenaire”, in *Puck*, No 2, Institut International de la Marionnette, Charleville-Mézieres, 1989.

Norman, S.J. (1996a). “The Art of Puppets”, in *Proc. IMAGINA*, Bry-sur-Marne, INA—Centre National de la Cinématographie, 1996.

Norman, S.J. (1996b). “Les objets de l'homme”, in *Puck*, No 9, Institut International de la Marionnette, Charleville-Mézieres, 1996.

Norman, S.J. (1996c). “Acting and enacting: stakes of new performing arts”/“Immersion and theater”, *Proc. International Symposium of Electronic Arts*, Montreal, 1996.

Norman, S.J. (1997a). “Acteurs de synthèse et théatres éléctroniques”, in *Le Film de théatre*, Paris, Editions du Centre National de la Rechèrche Scientifique, 1997.

Norman, S.J. (1997b). “Technology in the Performing Arts: Ways of Seeing, Ways of Doing”, in *Writings on Dance*, No 17, Victoria, Australia, 1997.

Norman, S.J. (1997c). “Les nouvelles technologies de l'image et les arts de la scène”, in *Théatre Public*, No 127, Paris, 1997.

Pereverzev, S.V., Loshak, A., Backhaus, S., Davis, J.C. and Packard, R.E. (1997). “Quantum oscillations between two weakly coupled reservoirs of superfluid 3He” *Nature* 388, pp 449–451.

Phillips, Cary B., Badler, Norman I., and Granieri, John (1992). “Automatic viewing control for 3D direct manipulation”. *1992 Symposium on Interactive 3D Graphics*, pages 71–74.

Pitas, I. (1993). *Digital Image Processing Algorithms*, Prentice Hall, ISBN: 0-13-145814-0, 1993.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. and Carey, T. (1994). *Human-Computer Interaction*, Addison Wesley, ISBN: 0-201-62769, pp. 281–282, 1994.

Pycock, J. and Bowers, J. (1996). “Getting others to get it right”. *Proc. CSCW'96*, 1996.

Rauterberg M., Bichsel M., Fjeld M. et. al. (1998), “BUILD-IT: A Planning Tool for Construction and Design”, Video Proc. + Summary CHI'98.

Reynard, G. T., Benford, S. D. and Greenhalgh, C. M. (1998). “Awareness Driven Video Quality of Service in Collaborative Virtual Environments”, Proc. ACM CHI’98, Los Angeles, April 1998, ACM.

Roads, C. (1988). “Introduction to Granular Synthesis.” *C. M. J.* Vol. 12, No. 2, pp 11–13.

Rodden, Tom (1996). “Populating the application: A model of awareness for cooperative applications”. In Ackerman 1996, pages 87–96.
http://www.acm.org/pubs/articles/proceedings/cscw/240080/p87-rodden/p87-rodden.pdf.

Sandor, Ovidiu, Bogdan, Christian, and Bowers, John (1997). “Aether: An awareness engine for CSCW”. In John A Hughes, Wolfgang Prinz, Tom Rodden, and Kjeld Schmidt, editors, *Proceedings of the Fifth European Conference on Computer-Supported Cooperative Work*, pages 221–236, Dordrecht, Boston, London, September 1997. Kluwer Academic Publishers.

Scaletti, C., and Craig, A. B. (1990). “Using sound to extract meaning from complex data.” In E. J. Farrel (ed.), *Extracting meaning from complex data: Processing, display, interaction 1259*, pp 147–153. SPIE

Seligmann, D. Duncan and Feiner, S. (1991). “Automated Generation of Intent-Based 3D Illustrations”, in *Proc. SIGGRAPH’91, Computer Graphics*, 25(4), July 1991.

Smith, Gareth (1996). “Cooperative virtual environments: lessons from 2D multi-user interfaces”. In Ackerman 1996, pages 390–398.
http://www.acm.org/pubs/articles/proceedings/cscw/240080/p390-smith/p390-smith.pdf.

Snowdon, Dave, Greenhalgh, Chris, and Benford, Steve (1995). “What you see is not what I see: Subjectivity in virtual environments”. In *Framework for Immersive Virtual Environments*, pages 53–69, December 1995.

Stary C (1996). “Interaktive Systeme. Software-Entwicklung und Software-Ergonomie”, Vieweg Informatik/Wirtschaftsinformatik, 1996.

Suzuki, G. (1995). “InterSpace: Toward Networked Reality of Cyberspace”, *Proc. Imagina ’95*, pp. 26–32, 1995.

Times (1998). “TV from another planet: something virtually different”, *The Times* (Interface section), October 7th 1998.

Turner, Russell, Balaguer, Francis, Gobbetti, Enrico, and Thalmann, Daniel (1991). “Physically-based interactive camera motion control using 3D input devices”. In *Proceedings Computer Graphics International '91*, 1991. http://ligwww.epfl.ch/~thalmann/papers.dir/CGI91.pdf.

Ullmer B., Ishii H. and Glas D. (1998). “mediaBlocks: Physical Containers, Transport, and Controls for Online Media”, *Proc. SIGGRAPH'98*, ACM 1998.

Underkoffler J. and Ishii H. (1999). “Urp: A Luminous-Tangible Workbench for Urban Planning and Design”, *Proc. CHI'99*.

Vernon, D. (1991). *Machine Vision: automated visual inspection and robot vision*, Prentice Hall, ISBN: 0-13-543398-3, 1991.

Walker, G. R. (1997). “The Mirror—reflections on Inhabited TV”, *British Telecommunications Engineering Journal*. 16 (1), pp. 29–38, 1997

Ware, C. and Osborne, S. (1990). “Exploration and Virtual Camera Control on Virtual Three-Dimensional Environments”, *Proceedings of the 1990 Symposium on Interactive 3D Graphics*, Special Issue of *Computer Graphics*, Vol. 24, pp. 173–183, 1990.

Wellner P (1993). “Interacting with Paper on the DigitalDesk”, *CACM*, Vol. 36, No. 7, July 1993, pp 87–96.

Xiao, Dongbo and Hubbold, Roger (1998). “Navigation guided by artificial force fields”, *Proceedings of CHI ’98*. pages 179–186.