Designing a VR Interaction Authoring Tool using Constructivist Practices

Cara Winterbottom
Collaborative Visual Computing Laboratory
University of Cape Town
South Africa
cwinterb@cs.uct.ac.za

Edwin Blake
Collaborative Visual Computing Laboratory
University of Cape Town
South Africa
edwin@cs.uct.ac.za

ABSTRACT
This paper describes the process of designing an authoring tool for virtual environments, using constructivist principles. The focus of the tool is on helping novice designers without coding experience to conceptualise and visualise the interactions of the virtual environment. According to constructivism, knowledge is constructed by people through interactions with their social and physical environments. Major aspects of this theory are explored, such as multiple representations, reflexivity, exploration, scaffolding and user control. Its practical application to the design of the tool is then described.

Categories and Subject Descriptors
D.2.2 [**Software Engineering**]: Design Tools and Techniques—*user interfaces*; H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems—*artificial augmented and virtual realities*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*evaluation/methodology, graphical user interfaces(GUI), screen design*

General Terms
Design, Theory, Human Factors

Keywords
virtual reality, constructivism

1. INTRODUCTION
This paper describes the process of designing a tool using constructivist principles. Specifically, these principles have been used in the design of an interaction authoring tool for virtual environments.

One of the main difficulties in VE design and creation is the conceptualisation of the possible interactions within a VE: user interactions with objects and the environment; and object interactions with each other and the environment. Because of the nature of VR, a linear script of the interactions that will occur in a VE is not possible. There are multiple possibilities and combinations, each of which must be considered and examined when a VE is designed. Even for experienced designers, conceptualising all of the possible interactions is a difficult task. Large design documents are produced, which detail the interactions and how they relate. These usually contain various visual representations, such as flowcharts, storyboards and floor plans, which help to organise the sequence of interactions. These designers have various techniques and ways of thinking that they can apply to the problem. For those who are inexperienced in design, creating interactions and visualising them in a complete and consistent sequence is even more daunting: they lack the practical experience and knowledge that would allow them to handle the complexity. The aim in designing this tool has been to try to reduce the complexity of this task for new designers. It must be mentioned that the tool rests on the assumption that actors and entities for the VE are created in a separate modelling package, along with their animations. This tool is only used to visualise and create the interactions *between* these entities.

The use of constructivist principles in the research and design of this tool was considered appropriate, as constructivism is a theory of knowledge that directly examines complexity and how it can be managed. According to constructivism, knowledge is constructed by the individual through interactions with the environment and social interactions[12]. How constructivism addresses complexity and other issues, such as multiplicity, exploration and reflexivity, is discussed in the following section. This is followed by a discussion of its practical use in the design of the tool. Finally, a simple example will be provided.

2. ASPECTS OF CONSTRUCTIVISM
Constructivism has its roots in cognitive science, particularly the developmental psychology of Piaget, the sociohistorical psychology of Vygotsky and semiotic interactionism[4]. There are many variants of constructivism, which emphasise different construction processes[5, 19]. However, there are important similarities between these versions of constructivism. They all focus on activities rather than on objects. These activities are creative and self-reflexive. It does not make sense to talk of knowledge of an objective

reality, as all knowledge is dependent on how we construct it. Therefore, coherence and viability are valued over an abstract notion of truth. Versions of constructivism also all focus to some extent on interaction, either within an individual in terms of the interaction between existing knowledge structures and new information, or between individuals in a society, jointly constructing knowledge[16]. Five aspects of constructivism that have directly influenced the design of this tool will now be described.

2.1 Hypermedia and the Labyrinth

As mentioned above, constructivism is related to semiotic interactionism. Semiotic interactionism focuses on the interaction between symbol and thought in concept development. The semiotician, Lucinda Leão[10] explores the concept of the labyrinth, as a model for hypermedia and their use. Hypermedia systems are complex. Therefore, each node should be clear, coherent, ordered and most of all, simple. The complexity comes from the links between these simple nodes. This idea is echoed in programming and user interface theory, where simplicity is valued, but complexity should also be possible[15, 11]. A specification of the interactions of a VE may be compared to a hypermedia system, where the nodes are atomic interactions which can have various consequences depending on time, space and state. An atomic interaction is here defined as the most basic interaction that happens between two or more entities in a VE, for example one actor greeting another actor. The various consequences of an interaction are the links which lead to other possible interactions. Creation of atomic interactions should be as simple as possible. The complexity is introduced when these atomic parts are related to others.

2.2 Multiplicity

Constructivist practice encourages multiplicity: multiple perspectives on concepts, multiple representations and multiple realities. Tools and environments should help the user to interpret the multiple perspectives of the world[12]. Multiple visualisations also help to control complexity, in the sense that it is broken down by focussing on different aspects of it. This corresponds to visual language theory and practice[9, 17], where different views are known to be appropriate for different kinds of data and tasks. Various existing design systems also promote the use of multiple representations, which are often juxtaposed to allow users to think in different ways about their creations, for example [14, 1, 6]. Multiple perspectives also allow a more complete view to be gained of the entire interaction sequence. By reflecting on representations, new possibilities or contradictions emerge. Users can learn to see the connections between the different interactions that they have created. In this way, multiplicity fosters conceptual interrelatedness, as different views can highlight how concepts relate to each other. For instance, a time-based view will highlight time-sequences and a spatial view of the same environment will highlight how the space is being used.

2.3 Reflexivity

Reflexivity is promoted, as individuals should think about their construction of the world in order to form viable theories. This principle exists in software engineering theory, for instance iteration in the waterfall model[18], and in design representations and visualisations[17, 2, 3]. Iteration assumes reflection so that the designer can improve on previous designs. Much research has been conducted on the use of external representations, indicating that they encourage reflexivity and promote a deeper understanding of the subject of the representations. Therefore, a design tool should provide mechanisms which allow the designer to step back from construction and think about how the user will operate the creation.

2.4 Exploration and Error Handling

Intuitions, successful experiences and observations play an important role in our actions. This is especially the case with novices in any area. Therefore, the potential for exploration is valued in any tool. Part of exploration is making mistakes and learning from them. Therefore, errors should be seen in a positive light and as a mechanism for the user to gain insight into her process, by providing feedback on the status of the design. General user interface and HCI design guidelines support this principle[15, 13]. Errors should be seen as part of the design and construction process, which the tool will help the user notice and correct. The tool should give the user feedback on the status of the construction, for example where there are holes in the sequence of interactions that have been entered into the tool. In this way, the user is informed and able to correct errors herself.

2.5 User Control and Scaffolding

In constructivism, users are provided with opportunities and incentives to build knowledge up, rather than having it given to them. In this way the user is in control. Schneiderman states that users should have control, which gives them feelings of success and competence. This encourages exploration[15]. As design is an ill-structured problem space with no predefined goals and constraints[3], this principle is even more important. In order that users can accomplish their specific design objectives, a design tool should provide them with as much freedom of expression as possible. Therefore, the tool should be passive and responsive rather than active and requiring response. Guidance can be provided in the form of scaffolding. This is the process of guiding the user from what is presently known to what is to be known, so that she can perform tasks that would normally be slightly beyond her ability. For instance, for help on design, a tutorial will be more appropriate than a wizard, so that the user collaborates with the tool to achieve the task, and also learns about the design process.

3. THE INTERACTION AUTHORING TOOL

A creation environment that supports multiple perspectives or interpretations of the reality that is being created encourages reflection and structuring. It provides an engaging problem-manipulation space, where users can directly manipulate or explore objects and activities and gain feedback through changes in the appearance of the representations. Four important aspects of the interaction authoring tool are described below, showing how they relate to the constructivist principles highlighted above.

3.1 Simple Input

The information that the user must enter for each part of an interaction in the tool has been designed to be as simple as possible and on a per-object basis. The complexity of the

process and the power of the tool lie not in the power of the atomic parts of interaction, but in how these relate to each other. This connects to the hypermedia concept of simple nodes and complex links. The nodes are the atomic interactions and the links are the relations between these. Because it is simple to enter atomic interactions, this should encourage exploration, as the time investment and consequences of adding new interactions are minimal.

3.2 Multiple Visualisations of the Interaction Sequence

In order to help the user conceptualise and plan the complex relations between interactions, multiple visualisations of the created set of interactions are provided. Each of these emphasises particular aspects of the sequence. The currently considered visualisations for this tool are as follows:

- *Timelines for time sequencing* — Timelines have been shown to reduce errors in temporal ordering[1, 6]. These representations serve a dual purpose in the tool. They can be created to specify a time-based sequence of events and actions. Any created timelines can also be viewed to understand how time is used in the VE. While each timeline belongs to an object, many objects can be represented on a single timeline, so that their interaction can be captured. The only object that cannot be represented is the user, as his actions are not predictable.

- *Floorplans for spatial sequencing* — Floorplans are used in architecture and engineering CAD packages to represent space. In the same way, they can be used to describe the usage of space in a VE, i.e. where objects have been placed, movement paths of actors and spatial triggers of interaction.

- *Sequence diagrams, which are modified statecharts, for narrative sequencing* — Each active object in the VE can be represented by a statechart, although the most useful statechart to view will probably be that of the user. The states are points from which interactions are possible and the links are actions that provide transitions to new states. This provides a connection from the atomic interaction to the sequence. Harel developed state charts for designing and maintaining complex reactive systems[7, 8]. A reactive system is defined as one which is heavily based on reactions to discrete occurrences. Statecharts allow complex information to be visualised in a manageable way, through the use of superstates and broadcast communication. A specification of the interactions of a VE may be described as a complex reactive system, where the discrete occurrences are triggering events, which can have various consequences depending on time, space and state. The statechart visualisation will be able to represent the reactions of the VE.

- *3D view for a visualisation of the end-product* — The 3D view allows the user to see the created VE as the end-user will see it. This gives a visualisation of the 3D space of the world. This view will be described more completely in the next section.

These representations are all related, so that if an object is highlighted on one of them, for example the floorplan, the same object will be highlighted where it appears on the other representations, for example the 3D view.

3.3 Run Mode of VE

In order to encourage reflexivity about the design process, the various visualisations can be viewed simultaneously. At any point, the interactions already created can be run as if the VE were being played by the end-user. During the run, a highlight moves through each open visualisation to indicate where the action is taking place (in terms of time, space or sequence). In this way, the tool will provide scaffolding to help novice designers to make the connections from the atomic interactions that they have entered, to the sequences of interactions that result. This functionality will also relate what the user experiences to the design. The 3D window allows the designer to run the VE as a designer in fly-by mode and as a player, where the VE is experienced exactly as the end-user would experience it. This fosters reflexivity in allowing the designer to step back from constructing the VE and think about the paths that a user might follow.

3.4 Errors as Design Process

Errors are viewed as part of the design process. This means that they appear to the user as a description of status, rather than as a pop-up error. They should be non-intrusive, like the warnings in a compiler which indicate atypical input. The tool will be aware of holes in the interaction sequences, and inform the user of where these are. For example, actions that are never triggered and locations that are not used sufficiently. In this way, the user is informed and able to correct errors herself, which promotes user control of the process and makes exploration more viable.

4. A SIMPLE EXAMPLE

A simple example will be used to highlight certain aspects of the tool mentioned above. Suppose that the designer wanted to design and create a scene where a Joker tells a joke to two Listeners and the user. The designer might conceptualise the actions as follows: when the user is in the same location as the Joker, the Joker tells a joke. During the joke, the Joker and Listeners have minor animations, such as head nodding. When the Joker ends the joke, the Listeners laugh. If the user is still there and looking at the Joker, the Joker will say: "I'm afraid that is the only joke I know." If the user is still there and not looking at the Joker, the Joker will say: "I don't think it's a good joke either." If the user is not there, the Joker will say: "That guy has no sense of humour."

While entering the interactions, the designer realises that all of the Joker and Listener actions are predictable after the user approaches, until the end of the joke. Therefore, he enters this part of the sequence on a timeline. A possible timeline is displayed in Figure 1.

As you can see from the diagram, the designer is able to place all three actors on the timeline and sequence their actions. The interface to the timeline uses direct manipulation to make the addition of actions or objects simple. If the VE is running and the timeline is being played, a highlight will move across it, indicating where in the time sequence the

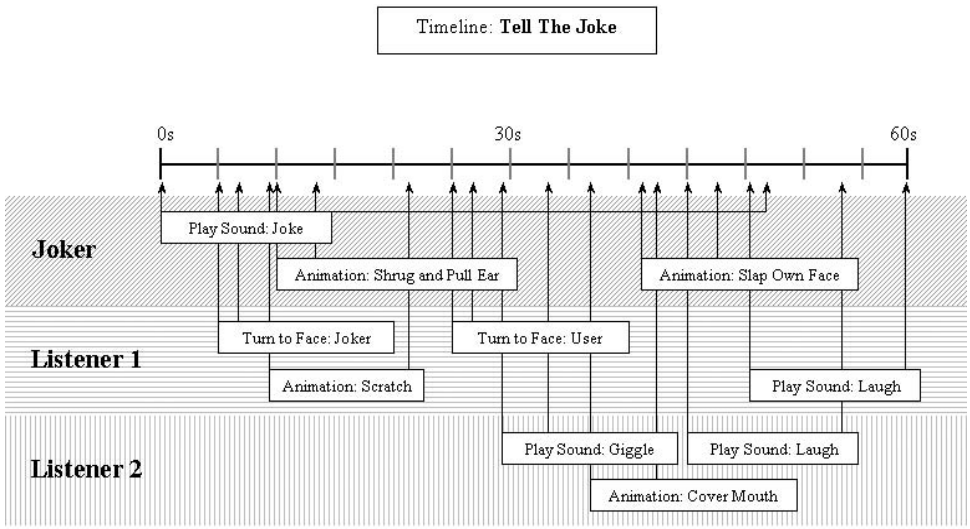

Figure 1: A Timeline of the sequence where the Joker tells a joke in the VE, indicating all of the actions of the joker and listeners during this period.

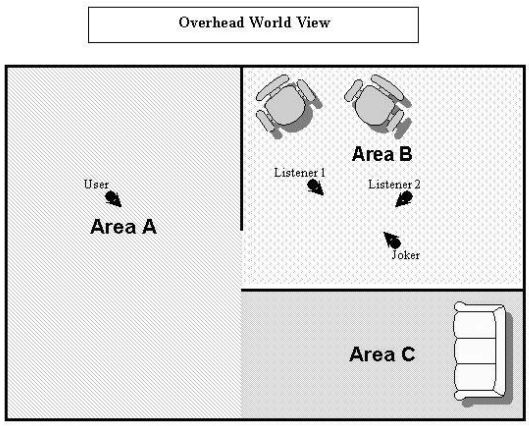

Figure 2: A Floorplan of the VE, indicating the positioning of the Joker, Listeners and user, as well as the different triggering Areas that have been defined.

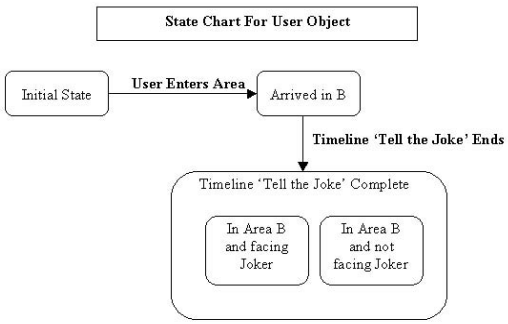

Figure 3: A Statechart of the user interactions in the VE.

action is happening. In order to keep the representation simple, branching timelines are not used: if there is the possibility of user interaction, the timeline will break at that point.

Having created the interactions, the designer may want to see how the space is used in the VE. Therefore, he will view the floorplan that has been set up. A possible floorplan for the VE is displayed in Figure 2.

The floorplan displays spatial relationships and contains an accurate representation of the geometry of the VE. The floorplan can be layered to reduce the complexity of the diagram, and layers can be toggled to portray different spatial aspects. For instance, gridlines, triggering areas, waypaths and objects can all be viewed on the floorplan, or turned off. The layers can also be used to view floorplans of multiple-level VEs. If the designer examines the floorplan, he will notice that he has not specified any interactions that will occur if the user enters Area C. Therefore, he knows to add further atomic actions which cover this eventuality. If the user clicks on an object in the floorplan, the object will also be highlighted in the timeline and the 3D view.

The designer may also want to see the reactions to user interactions that have been set up. Therefore, he will view the sequence diagram for the user. A possible sequence diagram for the user is displayed in Figure 3.

As you can see from the diagram, each state indicates a position where user interaction will have consequences. The states are labelled with the conditions that led to the state.

The links indicate the interactions that place the user in the new state. Superstates can be used when one interaction leads to multiple states. For instance, the state labelled "Timeline 'Tell the Joke' Complete" is a superstate, because new interactions involving the user are only possible when the timeline is finished. Within this superstate, two simple states are displayed. These correspond to the reactions that depend on the user interaction. Note that the state where the user has left the area is not shown, as the Joker's interaction does not affect the user in this case. If the designer examines the sequence diagram, he will see that there is not very much possibility of interaction set up for the user. Therefore, he might decide to add further interactions to make the VE more interesting.

5. CONCLUSIONS

Constructivism seems highly appropriate for the design of an authoring tool for novices, as it opens up many possibilities for how design might be accomplished and learned. Patterns of meaning are often discovered as a previously unseen truths. The elements may have been there, latent, but they were not visible. Constructivism is about organising and understanding the complexity of the world around us, so as to build knowledge. Because perception is incomplete, multiple perspectives may provide the possibility of new knowledge, which the user had not been able to conceptualise before. However, there is no indication in the literature that constructivism has been used in the VE authoring context before. Therefore, this project includes a plan to test the tool with novice designers, in order to provide empirical evidence of the success of constructivism for novice VE designers.

6. ACKNOWLEDGMENTS

Thanks to the CAVES project of the Innovation Fund of the NRF and to the University of Cape Town for funding of this research. Thanks also to the Methodology Group for insights.

7. REFERENCES

[1] O. Balet, P. Kafno, F. Jordan, and T. Polichroniadis. *Virtual Storytelling: Using Virtual Reality Technologies for Storytelling*, chapter : The VISIONS project, pages 90–99. O.Balet, G.Subsol and P.Torguet (Eds.). Springer-Verlag, 2002.

[2] A. Blackwell. Diagrams about thoughts about thoughts about diagrams. *Reasoning with Diagrammatic Representations II: Papers from the AAAI 1997 Fall Symposium, Tech Rep FS-97-03*, pages 77–84, 1997. M. Anderson (Ed.).

[3] C. Eastman. *Design Knowing and Learning: Cognition in Design Education*, chapter 1: A Constructivist Approach to Teaching. Elsevier Science, 2000.

[4] C. T. Fosnot. *Constructivism: Theory, Perspectives and Practice*, chapter 2: Constructivism: A Psychological Theory of Learning, pages 8–33. Teachers College Press, 1996.

[5] K. J. Gergen. *Constructivism in Education*, chapter 2: Social Constructionism and the Educational Process, pages 17–40. Lawrence Erlbaum Associates, 1995. Papers presented at the Alternative Epistemologies in Education Conference (Feb 1992).

[6] K. Harada, E. Tanaka, R. Ogawa, and Y. Hara. Anecdote: a multimedia storyboarding system with seamless authoring support. *ACM Multimedia*, pages 341–351, 1996.

[7] D. Harel. On visual formalisms. *Communications of the ACM*, 31(5):514–530, May 1988.

[8] D. Harel, H. Lachover, A. Naamad, A. Pnueli, M. Politi, R. Sherman, A. Shtull-Trauring, and M. Trakhtenbrot. STATEMATE: A working environment for the development of complex reactive systems. *IEEE Transactions on Software Development*, 16(4):403–414, April 1990.

[9] R. Horn. Visual language and converging technologies in the next 10-15 years (and beyond). Presentation at the National Science Foundation Conference on Converging Technologies (Nano-Bio-Info-Cogno) for Improving Human Performance, December 2001.

[10] L. Leão. The labyrinth as a model of complexity: the semiotics of hypermedia. *Computational Semiotics for New Media (COSIGN)*, September 2002.

[11] G. Marsden. *Designing graphical interface programming languages for the end user*. Phd thesis, Department of Computer Science, Stirling University, London, January 1998.

[12] E. Murphy. Constructivism: From philosophy to practice. URL: http://www.stemnet.nf.ca/˜ elmurphy/emurphy/cls.html, Summer 1997. Accessed: 25 March 2004.

[13] D. Norman. *The psychology of everyday things*. Harper Collins Publishers, USA, 1988.

[14] T. Schiphorst, T. Calvert, C. Lee, C. Welman, and S. Gaudet. Tools for interaction with the creative process of composition. *CHI '90 Proceedings*, April 1990. ACM Press.

[15] B. Schneiderman. *Designing the user interface: strategies for effective human-computer interaction*. Addison-Wesley Publishing Co., Massachusetts, 2nd edition, 1992.

[16] J. Shotter. *Constructivism in Education*, chapter 3: In Dialogue: Social Constructionism and Radical Constructivism, pages 41–56. Lawrence Erlbaum Associates, 1995. Papers presented at the Alternative Epistemologies in Education Conference (Feb 1992).

[17] N. Shu. *Visual Programming*. Van Nostrand Reinhold Company, New York, 1988.

[18] I. Sommerville. *Software Engineering*. Addison-Wesley Publishing Co., Massachusetts, 4th edition, 1992.

[19] E. von Glasersfeld. *Constructivism: Theory, Perspectives and Practice*, chapter 1: Introduction: Aspects of Constructivism, pages 3–7. Teachers College Press, 1996.