Using Visualizations to Support Design and Debugging in Virtual Reality

Cara Winterbottom, Edwin Blake, and James Gain

Collaborative Visual Computing Laboratory,
Department of Computer Science, University of Cape Town,
Private Bag X3, Rondebosch 7701, South Africa
{cwinterb, edwin, jgain}@cs.uct.ac.za

Abstract. We present a visualization system that helps designers conceptualise interactions in a virtual environment (VE). We use event-condition-action triads (triggersets) for specifying interactions, and provide multiple visualizations: sequence diagrams, floorplans and timelines. We present a two part study: sequencing VE interactions accurately and debugging mistakes. Subjects were divided into two groups: one received visualizations and triggersets and the other (a control group) received triggersets only. The visualization group described 72.5% of the sequence correctly on average, compared to 56.4% by the non-visualization group. The visualization group also detected more than twice as many errors as the control group. The visualization group worked well with multiple, linked windows to create an understanding of the design. Floorplans were most useful for an overview, timelines for understanding specific sequences and sequence diagrams for sequencing and finding mistakes.

1 Introduction

The field of Virtual Reality (VR) has all of the complexities of 3D world creation as well as the difficulties of providing interactions within each world, which are compounded because VR presupposes an independent user. Interactions are the relationships set up between the user of a VE, its objects and the environment itself. The design of interactions is difficult for several reasons: (1) They happen over time for an indeterminate duration, so the design cannot be viewed statically. (2) They happen for various entities, so that each entity or group of entities may participate in a different set of interactions. This leads to a combinatorial escalation of possibilities for interaction. (3) At least some of the interactions will be determined by what the user does, which cannot be pre-specified. Therefore, the designer must deal with a significant amount of uncertainty about how the end result will be experienced. (4) Interactions include actions that are so commonplace to us that we do not naturally think about them, like avoiding obstacles and facing the person to whom we are talking. They require significant detail to define completely. (5) Very often the VE will have a purpose or tell a story, which means that the user must be guided by the interactions and the

environment to achieve a goal. (6) The VE must be sufficiently reactive to the user's interactions to make the experience enjoyable and interesting.

In this paper, we describe the use of visualizations to support the design and debugging of interactions within a VE. We accomplish this principally by providing multiple visualizations: a *floorplan*, *timelines* and a *sequence diagram* of the flow of interactions for narrative sequencing. We also describe a study designed to test the effectiveness of these visualizations. In software visualization, which is related to our work, the outstanding questions include: with which problems are diagrams better than text, how do individuals differ in how they work with diagrams and what are the benefits of using multiple representations [1]? These underlie our research and motivate the study. We begin by providing some background in Section 2. In Section 3, our visualizations are described within their context, and in Section 4 we present our study and its results. We conclude in Section 5 with a discussion of the implications of our work.

2 Background

In this section we discuss previous research on program visualization and using visualizations for design and debugging. We also focus on the use of multiple visualizations. Visualization research has a long history, particularly in scientific and information visualization [2, 3]. In recent years, research has increasingly been conducted into the use of visualizations for a greater variety of problems [4]. For example, the use of visualizations to support programming tasks, such as debugging and control structure creation [5, 6]. Program visualization helps the designer to see the flow of control through a program[5]. To assist in the debugging task, Ko and Myers [7] developed Whyline, which provides visualizations of a program's runtime states in response to questions about what went wrong.

Visualizations can be used by experts during various design processes. General research on external representations suggests that they are useful in promoting reflexivity and a deeper understanding of their subjects [8, 1, 9, 3]. Eastman [9], in a survey of representations used in design, suggests that novices learn from viewing and working with external representations, so that in time these are internalised and become part of the designer's reasoning tools. Petre and Blackwell [10] found that during program design, expert programmers used various forms of mental imagery to think about a task. Commonalities in the imagery used included the fact that they were dynamic, but could be stopped and reversed; they had adjustable granularity; and they included simultaneous multiple images. Baldonado et al. provide guidelines for the use of multiple views in information visualization: they should be used when there is diversity of information, when different views elicit correlations or disparity in the information, or when complex data can be decomposed into manageable chunks. Providing multiple views fosters a deeper understanding of a problem when the distinct representations are understood as describing facets of the same idea [11].

3 A Linked, Multi-View Visualization System

There are various aspects to VE interaction design: the use of space, time and possible sequencing of interactions. These must all be considered in addition to the complexity added by the VE user's freedom of movement. We provide visualizations to help designers understand interactions. The use of multiple visualizations is intended to break up the complexity of interaction design by focusing on different aspects of it. Three visualizations are used: (1) a floorplan visualization of the space of the VE (see the left side of Figure 1a); (2) timelines, which are both a construction and a visualization tool (see Figure 2); (3) a sequence diagram for visualizing and debugging the sequence of interactions (see Figure 1b). The visualizations and how they link together are described further below. For programming the interactions, we use an event-action paradigm. Trigger-condition-action triads, which we refer to as triggersets, are used. The visualization system is embedded in this authoring system. A typical process of using our visualizations is as follows: The designer specifies objects and the environment. Time-based sequences of actions can be entered using timelines. The system generates a floorplan from the object positions and locations in the environment that have been specified. As soon as the designer has entered a few triggersets and object details into the system, she can view the sequence diagram generated by these interactions. Thus, the system allows for incremental programming, as the status and consequences of what has been programmed can be checked at any point by examining the visualizations.

Floorplan Floorplans have been shown by our own and other experiments to be very useful in the design of 3D worlds [12, 13]. They are also used successfully in engineering and architecture to represent space. Floorplans are essentially maps, which allow complex 3D worlds to be viewed in a more simple 2D way [3, 14]. In particular, lines of sight, locations and object positions can be easily viewed. In our floorplan, the VE space may be divided into rectangular regions called locations, which can be used in triggers or conditions without specific coordinates. The floorplan automatically displays the positioning of any object that has been given spatial coordinates and indicates its orientation. Any proximity triggers set up in relation to objects are shown on the floorplan. The floorplan is marked with a grid in the units of the world so that distances can be estimated. A compass is used to indicate direction and assist the designer in understanding the rotation system for objects. Designers can interact with the floorplan using direct manipulation. When an object's icon is selected, it is highlighted and the object details are displayed. The floorplan can also be layered to reduce complexity and to show different levels of a 3D world, if necessary. Figure 1a is an example of a floorplan used in the system.

Sequence Diagram Our sequence diagrams are inspired by Harel's statecharts [15], which were developed for designing complex reactive systems. They are generated directly from the triggersets and follow the flow of data through the VE.

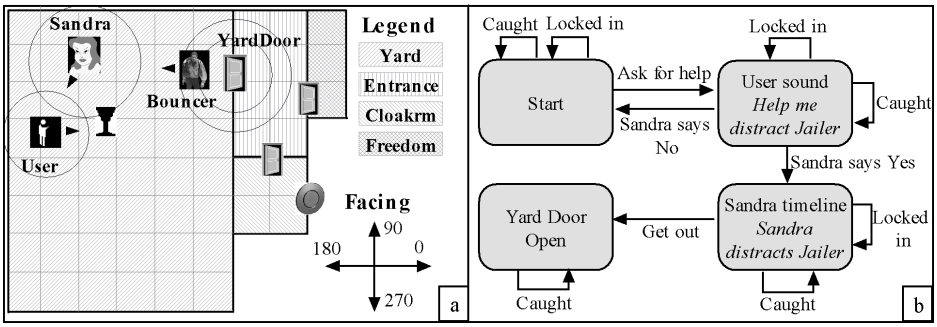

Fig. 1. Floorplan and Sequence Diagram. (a) is a floorplan of a VE. The position and orientation of each object is indicated by an icon and directional arrow, respectively; circles around each object correspond to proximity triggers that have been set up. (b) is a corresponding sequence diagram showing states and links. A state is a point from which user interactions will have consequences, and a link is a triggerset that can be executed from its originating state

States are identified where specific interaction possibilities exist. Each state specifies the current conditions in the environment that allow triggersets to execute. The states are linked by arrows, which correspond to triggersets' execution. If a triggerset does not change the current interaction possibilities, its arrow leads back to the same state. When the user clicks on a state, the arrows that leave it and the states to which they connect are highlighted. A description of the state also pops up. When the user clicks on an arrow, all other arrows referring to the same triggerset are highlighted and a description of the triggerset pops up. In this way, designers can step through their interactions and think about how the sequence fits together. For debugging interactions, the sequence diagram allows designers to see the effects of the triggersets; where triggersets have unexpected consequences and which triggersets never execute. A sparse diagram will indicate a lack of interactions provided in the VE. Because the sequence diagram itself is not linear, designers are encouraged to view the interactions in a non-linear way. Figure 1b provides an example of a sequence diagram.

Timelines Timelines are a well understood formalism and have been shown to reduce errors in temporal ordering [11, 12]. Because VE authoring depends on user interactions, our timelines do not represent VE time from start to finish. Instead, they represent parts of the VE where a sequence of actions will happen in known time. The actions are grouped on the timeline and then treated as a single action. For example, if a story is told in a VE, the sound file and actions of storyteller and listeners must all be coordinated. The timeline can be used to sequence the storytelling and reactions with precision and efficiency. The story timeline can then be executed as a single action. Each row on the timeline corresponds to an object and contains any actions that the object performs. The length of the action is automatically calculated (e.g., an animation's basic length

is read from its file and then multiplied by repetitions) and visualized on the timeline. Any action on a timeline can be selected to uncover more details about it. Figure 2 displays a typical timeline from the system.

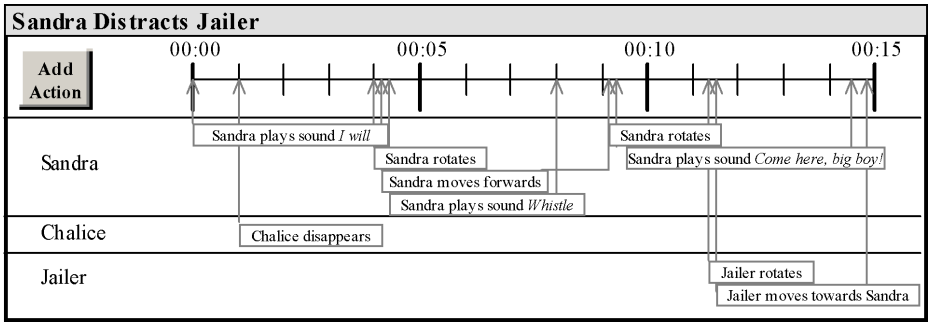

Fig. 2. Timeline showing objects involved and their actions. These can be selected to elicit more detail about each action

Linking Between Visualizations All our visualizations are linked to each other. The objects and locations referred to in each visualization connect them to one other and to the triggersets, so that they can be cross-referenced. Timelines are described in the sequence diagram. For example, Figure 1b displays a sequence diagram for a VE where the state, *Sandra Timeline Sandra Distracts Jailer*, describes the state of the VE while the timeline, *Sandra Distracts Jailer* is playing. From this diagram, one can see that the triggerset *Sandra Says Yes* begins the timeline and that three triggersets can be activated by the timeline or actions contained in it. If the designer selects this state, a detailed description of the timeline is provided and if the designer selects any of the triggersets connected to the state, a detailed description of each is given. The designer can use the name of the state to open the actual timeline (Figure 2). The names of the links can be used in a similar way to find the actual triggersets to which they correspond. The names of objects and locations in the sequence diagram, timelines and triggersets can also be used to find objects on the floorplan (Figure 1a). Therefore, by working through the visualizations in combination, the designer can gain an overview of the sequence of triggersets and how they interact with each other, as well as a detailed description of each.

4 Visualization Study

An exploratory study was conducted to test the effectiveness of our visualizations. For this initial study we wanted to find out how effectively the visualizations and triggersets were used by people in understanding a sequence of 3D

interactions and in debugging errors in a sequence. This aim has four parts: (1) To assess how subjects were able to describe the possible sequence of interactions in a VE. (2) To investigate the extent to which subjects could identify errors in a second set of interactions. (3) To investigate which visualizations subjects preferred to use in different contexts. (4) To assess how well subjects work with multiple visualization windows. We included a control group, who did not receive visualizations, so that we could compare the performance of subjects who did receive visualizations with that of subjects who did not. Because of the exploratory nature of our work, we decided to use participant observation for our study [16]. We observed subjects working on simple problems using our system. After the tasks were finished, the researcher discussed the subjects' experiences with them in a structured interview.

Part of the study dealt with identifying problematic interactions. Most mistakes, in our experience, fall into one or more of four general categories: *timing* errors arise from the time it takes the user or other objects to complete actions; *spatial* errors arise from the way space is used, in terms of orientation and location of objects; *sequencing/logical* errors arise from problematic ordering of interactions and the way they interrelate; and *implicit assumption* errors arise from the designer forgetting to state all behaviour explicitly. Examples of these errors are shown in Table 1.

4.1 Description of the Study

Our study had two parts, each in a separate VE. Visualization subjects were provided with a floorplan, timeline(s) and a sequence diagram of each. For the Sequence part of the study, a simple VE entitled *Bouncer Example* was used. The physical space of the *Bouncer Example* consisted of three locations. In addition to the User avatar, the VE contained a Bouncer (or guard), a Door, a Bell and a Chalice. The goal was to distract the Bouncer away from the Door that he was guarding, so that the user could open the door and grab the chalice. Various triggersets were set up to describe the possible interactions. The aim of this part of the study was to see how well subjects could work out what might happen in the VE. Therefore, the triggersets were set up so that some could not execute without others having been triggered, so that there was a sequence.

For the Debug part of the study, a VE entitled *Jail Example* was conceptualised. The space consisted of four locations. In addition to the User avatar, the VE contained a Jailer, an object named Sandra, three Doors, a Push-Button and a Chalice. The goal of the VE was to use Sandra to distract the Jailer, so that the User could get into the location named Freedom. The floorplan and sequence diagram for this VE are shown in Figure 1. The *Sandra Distracts Jailer* timeline shown in Figure 2 is also from this VE. As in the Sequence part of the study, various triggersets had been set up describing the interactions that could happen. However, in this case, several mistakes had been introduced into the triggersets. The aim here was for each subject to identify as many potential errors as possible. This was intended to test how well subjects could debug in-

correctly programmed interactions. The mistakes and the error categories from which they draw are indicated in Table 1.

Table 1. Interaction Programming Mistakes Introduced into the Visualization Study

Mistake	Description	Error Category
Jmove	Jailer does not move far enough and so proximity trigger is not executed	Spatial/Timing
Fopen	No way specified to open a door behind which is a button to open the door to Freedom location	Implicit Assumption / Spatial
CLcon	Caught and Locked In triggersets both execute and conflict	Sequencing and Logical
YNcon	Sandra Yes and Sandra No triggersets both execute and conflict	Sequencing and Logical

4.2 Procedure

A sample of eleven graduate students from different disciplines was recruited. It was decided to use people with graduate degrees or equivalent working experience, as these corresponded to the target group of people who might work with VR. None of the subjects had any experience with graphics programming, although they had experience with graphics packages ranging from none to Illustrator and 3D Studio Max. Gender was evenly represented. Subjects were volunteers within these constraints and were paid a small amount for their participation. A control group of five subjects who would not receive visualizations was randomly selected from the larger sample.

Each subject experienced the study individually and was then interviewed. Before the study began, the subject was given an introduction in which interactions, the triggerset formalism and the visualizations (only for the visualization group) were introduced. Following this, subjects were given 20 minutes to examine the triggersets and visualizations for the Sequence part of the study. After this they were asked to write down what might happen in the VE, indicating any dependencies among the triggersets. Thereafter, the instructions and VE descriptions for the Debug part of the study were provided. Subjects were asked to specify any mistakes that they found and were also given 20 minutes to examine the triggersets for this part. When subjects had finished with both parts of the study, a structured interview was conducted. A code was developed for scoring the accuracy of the sequence descriptions of the Sequence part of the study. Points were given for correct sequencing, awareness of branches, awareness of object locations and how these might change, and awareness of pre-conditions on actions. An independent marker examined the scripts using the code, in order to ensure that the marking was unbiased.

4.3 Results and Discussion

All of the subjects found the triggerset formalism easy to use and understood how the triggersets worked. It was when they had to be ordered in a sequence that subjects had difficulties. Visualization subjects achieved a 72.5% average on correctly working out the Sequence part of the study, while the control group only achieved a 54.6% average. The different between means was just not significant ($F = 6.32$, $p < 0.09$). However, once an outlier in the control group was removed, its average dropped to 48.5% and the difference became significant ($F = 1.43$, $p < 0.002$). For the Debug part of the study, the visualization group noted twice as many errors as the control group (37.5% vs. 15%), but overall the number of errors noted was low. This result was not tested for significance, as most of the control group(3 of 5)found zero errors, which skewed their results. Subjects were limited to 20 minutes to debug the VE, which partially accounts for the poor results. Only one (visualization) subject found Jmove and CLcon (see Table 1 for a description of the errors). Two visualization subjects and one control subject found Fopen. But five visualization subjects (and two control subjects) found YNcon. We can examine the errors that were detected, broken into error types. The visualizations seem most helpful for sequencing errors, probably because the sequence diagram indicates possible sequences. Without this, the triggersets must be manually connected to each other, based on the result of each one being executed. Timing errors are almost impossible to find without some way to view the final product or step through the events.

Subjects found the floorplan most useful: to orient, give a concrete sense of the space and where the objects are in relation to each other: "Once you coordinate between the physical locations, you can see how you need to move." In fact, half of the subjects stated that they could not have reconstructed the sequence without the floorplan. Timelines were useful for noticing a predictable sequence: "Used the timeline for Bouncer Move to see how he went away." One subject stated, "I did not use timelines much to examine interactions because they were simple, but they would be very useful to make actions — work very nicely. For design, I like the timeline. It is important as both a visualization and a construction tool." Mistakes in the Debug part of the study were made more obvious by the sequence diagram: "The sequence diagram is useful for seeing how the triggersets relate, their order and what activates what." "Could walk anywhere, but the VE only reacts like the sequence." Even those who used the sequence diagram less stated that it was useful to "check up after your own analysis of the triggers". Subjects did state that they wanted more interactivity from the sequence diagram.

A typical method of working with the visualizations was to begin with the floorplan in order to gain an overview of the VE and its interactions. Subjects would then examine the triggersets and object descriptions, referencing the names and positions of objects and locations on the floorplan. During this process they would repeatedly go to the sequence diagram to find out how triggersets sequenced, or confirm their own analyses. When timelines were referred to, they would open them and check the objects and actions. In this way, sub-

jects used the linking between the visualizations effectively. Subjects all worked in different ways with the visualizations and triggersets. This justifies the flexibility of the tool in allowing for different work processes. There were similarities, though: Subjects all looked at multiple visualizations and most cross-referenced the visualizations constantly to work out what was happening in the VE. Only one subject mentioned problems with switching between multiple windows. They all found the visualizations clear and useful but found each one useful for different things, e.g. "I had all three (visualizations) open at the same time. Then, if you don't understand the sequence you can look at the timeline." And "The triggersets are basically just the details of the rest (the visualizations)". They all felt that the tasks would have been much harder without the visualizations. A few mentioned that the interactivity was very helpful.

Our study also indicated areas where we could improve the visualizations. Subjects underused the timelines, as they were more difficult to access. This means that *ways to access the visualizations must be made highly visible and simple.* People need to be able to access each visualization from the others, so that they do not overlook them and to make it easier to cross-reference. Subjects did not often drill down into triggersets and object details to find out necessary information, e.g., the angle of rotation for an accurate point of view. Therefore, *details that are not visualized must be made more obvious and easily accessible.* Subjects indicated that it would be helpful to group triggersets according to various criteria, such as locations in which they might execute. Users also need more help with working out the consequences of rotations and translations, in terms of where an object will end up after a sequence. *More support is needed to help users understand certain parts of 3D interaction design, such as spatial relationships.* The visualizations can guide users more in this regard.

5 Conclusion

We have described our experiences in providing visualizations to support the understanding of VE interaction design. In the study we conducted, visualization subjects consistently out-performed the control group. They understood the visualizations and worked well with them. They naturally used multiple interacting visualizations to build a complete idea of a complex design. Floorplans were used to gain an overview of the VE and interactions, timelines were used to understand the consequences of predictable sequences of events, and sequence diagrams were used to understand how the interactions worked together and to find mistakes. The interactivity that was added was also very positively received, although subjects would have preferred more. More work must be done on assisting users with debugging tasks, especially in terms of highlighting inappropriate or inconsistent interaction effects.

These results show that using visualizations to understand and debug VE interactions improves the authoring process. This is a key finding in helping to make VR more accessible as a creative medium. It also provides us with validation of the efficacy of our visualizations, which creates a solid basis for

future work. Our next step is to add a 3D view of the VE to complement the other visualizations. Then we will develop a run mode with a highlight which moves through all of the visualizations, indicating where the action is taking place. More interactivity will also be added between the 3D window and the other visualizations.

Acknowledgements

This research is funded by the South African National Research Foundation (GUN 2053402) and by the University of Cape Town.

References

1. Blackwell, A., Whitley, K., Good, J., Petre, M.: Cognitive factors in programming with diagrams. Artificial Intelligence Review **15** (2001) 95–113
2. Card, S., Mackinlay, J., Shneiderman, B.E.: Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann Publishers, California (1999)
3. Shu, N.: Visual Programming. Van Nostrand Reinhold Company, New York (1988)
4. van Wijk, J.: The value of visualization. In Silva, C., Groeller, E., Rushmeier, H., eds.: Proceedings of IEEE Visualization, IEEE (2005) 79–86
5. Romero, P., Cox, R., du Boulay, R., Lutz, R.: A survey of external representations employed in object-oriented programming environments. Journal of Visual Languages and Computing **14** (2003) 387–419
6. Myers, B.: Taxonomies of visual programming and program visualization. Journal of Visual Languages and Computing **1** (1990) 97–123
7. Ko, A., Myers, B.: Designing the whyline: A debugging interface for asking questions about program behaviour. In: Proceedings of CHI 2004. (2004) 151–158
8. Kavakli, M., Suwa, M., Gero, J., Purcell, T.: Sketching interpretation in novice and expert designers. In Gero, J., Tversky, B., eds.: Visual and Spatial Reasoning in Design, Australia, Key Centre of Design Computing and Cognition (1999) 209–220
9. Eastman, C.: New directions in design cognition: studies of representation and recall. In Eastman, C., McCracken, M., Newsletter, W., eds.: Design Knowing and Learning: Cognition in Design Education, Elsevier Science (2000)
10. Petre, M., Blackwell, A.: Mental imagery in program design and visual programming. International Journal of Human-Computer Studies **51** (1999) 7–30
11. Baldonado, M., Woodruff, A., Kuchinsky, A.: Guidelines for using multiple views in information visualization. In: Proceedings of AVI, ACM Press (2000)
12. Harada, K., Tanaka, E., Ogawa, R., Hara, Y.: Anecdote: A multimedia storyboarding system with seamless authoring support. In: ACM Multimedia, ACM Press (1996) 341–351
13. Coleburne, A., Rodden, T., Palfreyman, K.: VR-MOG: A toolkit for building shared virtual worlds. In Slater, M., ed.: Proceedings of FIVE (Framework for immersive virtual environments) Working Group Conference. (1995) 109–122
14. Tufte, E.: The Visual Display of Quantitative Information. Graphics Press, Cheshire, UK (1983)
15. Harel, D.: On visual formalisms. Communications of the ACM **31** (1988) 514–530
16. Banister, P., Burman, E., Parker, I., Taylor, M., Tindall, C.: Qualitative Methods in Psychology: A Research Guide. Open University Press, Buckingham, UK (1994)