Eliminating Design and Execute Modes from Virtual Environment Authoring Systems

Gary Marsden & Shih-min Yang
Department of Computer Science, University of Cape Town, Cape Town, South Africa
Email: gaz@cs.uct.ac.za, *Tel*: +27 21 650 2666, *Fax*: +27 21 689 9465

Abstract

In this paper we report on efforts to create a virtual environment authoring tool for novices. In particular we set out to eliminate separate design and execute behaviors from these tools. We present two alternative prototypes for achieving this and report on the results of a usability experiments comparing each environment.

Keywords- **Virtual reality authoring, modes, programming environments.**

1. Introduction

Within the Computer Science Department at the University of Cape Town we are embarked on building a software that supports novice users in creating desktop virtual environments. The most comparable system to our own is Alice[1], but whereas Alice is designed explicitly for children, our system is designed for domain experts (e.g. architect, teacher etc.) who do not have the necessary computer science skills to use existing systems such as DIVE[2] or Genesis[3].

The work for creating this tool is divided across various groups of researchers and programmers; in this paper we will concern ourselves only with the specific portion of the research conducted into the high level interface design.

In creating an interface to the authoring system, our intuition was that the biggest barrier to novices would be the programming interface. From our initial observations, however, we discovered that even before they reached the scripting level, users were confused by the whole notion of having separate design and execute modes. Users would click in frustration at objects, expecting them to react, only to be greeted with an attribute browser window.

It has long be known that modes in general are undesirable in a user interface, but this problem of separate design and execute modes has also been well reported in research relating to 2-dimensional interface authoring tools [4]. It seems that this duality, which is so obvious to programmers, confuses those not used to programming environments. This manifests in the form of mode confusion behaviors, like the appearance of the attribute browser, mentioned above.

In virtual environment authoring, we speculated that the impact could be even greater than in two dimensional interface builders.

Virtual environments are often touted as the last word in direct manipulation environments [5]. In a sense, there is no interface as the environment is a complete visualization of the entire system. In a really good virtual environment, the user should have a sense of ‘being there’[7], focusing more on the virtual environment than their physical environment. This notion of ‘being there’ is formally called ‘presence’ and is a very active research area in the virtual reality community. Environments which generate a strong sense of presence are obviously more effective than environments which do not (if the user does not relate and engage with the VE, then there is little point in creating it in the first place). One of the ways in which environments may inadvertently reduce presence is through the intervention of external events, or breaks-in-presence (BIPs) [13]. If we return to our user who is engaged in the activity of creating an environment, the need to switch out of the execute environment, to the design environment, to effect a change must surely introduce a break in presence. (The break would be further exacerbated if the user was wearing data gloves and a head mounted display). It would seem logical, therefore, to create an interface which allowed the user to alter the environment from within the environment itself. This will necessitate the removal of the “design” environment completely and somehow integrate that functionality into the “execute” environment.

2. Previous Research

By removing the design environment, we do not remove the need for separate modes of interaction. In the parlance of [4], we will always want to use things and other times want to mention them, regardless of being in a real world or a virtual one. How then should one add these two capabilities to a virtual world?

The goal of supporting in-environment use and editing is not entirely new. For example, the Worlds in Miniature (WIM) system allows the user to manipulate a facsimile of an environment whilst in that environment [6]. However, it forces splitting the display real estate between the original environment and the miniature copy. In addition, it is difficult to select, and manipulate objects especially for fine-grained manipulations as the entire environment is scaled down into a hand-held size [8]. Perhaps most crucially, the metaphor cannot maintain the sense of presence because working with two different worlds emphasizes that the world is artificial.

The “Voodoo Dolls system” was proposed and implemented by Pierce et al [9]. This technique allows the users to interact with objects that are beyond one’s physical reach. It supports direct manipulation of an object by creating miniature copies of the object. This method gives the user an illusion of interacting as if in real life. However, the miniature copies of objects have different properties when they are held on the user’s right or left hand. This feature might confuse novice users since they might not fully understand the concept of the metaphor. Furthermore, presence may be reduced as users are forced to interact with the ‘doll’ of an object, rather than the object itself.

We are looking for a more general solution which tries to mirror the user’s real world experience as closely as the virtual environment will allow.

3. Migrating Functionality

Returning again to the two dimensional interface world, the “tool” approach is one of the proposed solutions for the problem of “mention and use” in a single window[4]. The tool approach is derived from everyday experience by adopting the interaction methods used in real life. It uses the idea of direct manipulation mediated through some tool. Every interaction requires some kind of tool, e.g. a “paintbrush” is used to paint objects, and a “hand” tool is for grabbing objects. Different tools can be seen as different modes so the interaction takes on the form of a global mode.

An alternative to the tool approach is adopting a mode-per-object. This approach allows different objects to be in different modes, meaning that there will be multiple active modes in one environment. An example of using the “mode per object” approach is presented in our earlier work[10]. A screw is attached to every user interface component. To edit a particular component, one clicks on the screw which reveals an attribute browser. To use a component, one simply clicks on any part of its surface (other than the screw) as in a normal system – see Figure 1 for a typical sequence if interactions. Thus in one environment, some interface components are in design mode, while some are in execution mode. In this particular interface, the screw is shown screwed in to reflect an object in execute mode; or shown screwed out to reflect edit mode.

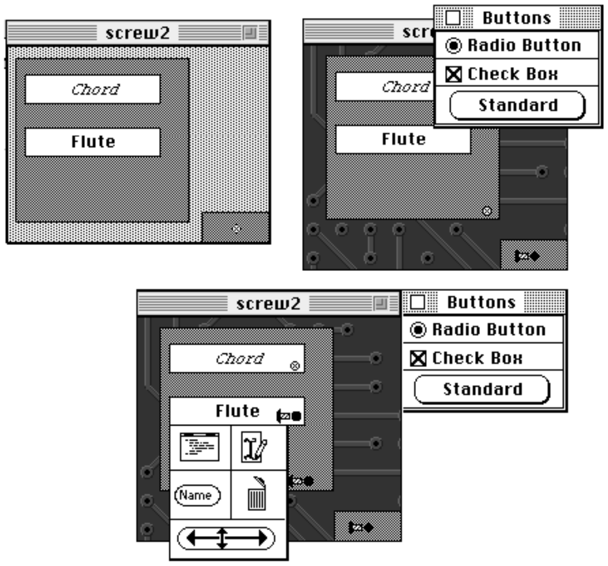


Figure 1. The screw in the bottom left is first removed to reveal more screws and a parts tray. Undoing the screw on an individual widget allows details of the widget to be edited

Certainly this approach more accurately reflects the real world where different objects can be in both modes simultaneously. The difficulty of this approach are visual cues to indicate state, as they consume space on the virtual objects. Additionally, these visual cues would diminish the sense of the presence as users are aware of the artificial visual cues which appear on the virtual objects – we cannot simply place screws on every object in a virtual environment.

Whilst the mode-per-object approach more closely mirrors real world behavior, the tools approach is more common in current software. Therefore, before committing to a particular design we set out to investigate if different ways of interacting with objects will cause users to behave differently.

4. Investigating Behavior

We hypothesize that the users will change their ways of working depending on the system they are using. For instance, with the tool approach, we expect the users to work in the fashion of “tool by tool” and users might work “object by object” in the mode-per-object system.

4.1 Tool system

In real life, workers normally carry a toolbox to the working site. When they want to modify or fix an object, they take out an appropriate tool from the toolbox. Once the job is done, they put the tool into the toolbox and walk to another place with the toolbox.

Most virtual environments are quite similar to real life in this respect, and therefore we felt that using the toolbox idea in virtual environments is appropriate. To implement the idea, a toolbox is provided in the virtual environment as a 3D object that can be opened and closed. In real life, the toolbox is not seen until it is actively sought out. It is difficult to implement this in the virtual environment because objects outside the users' view frustum are difficult to access. Instead, we place the toolbox in a fixed location related to the user's viewpoint. Thus the users know where to find the toolbox when it is needed, and time is reduced in searching the entire virtual environment for the toolbox if it is located at a fixed position. Figure 2 shows the tool environment.

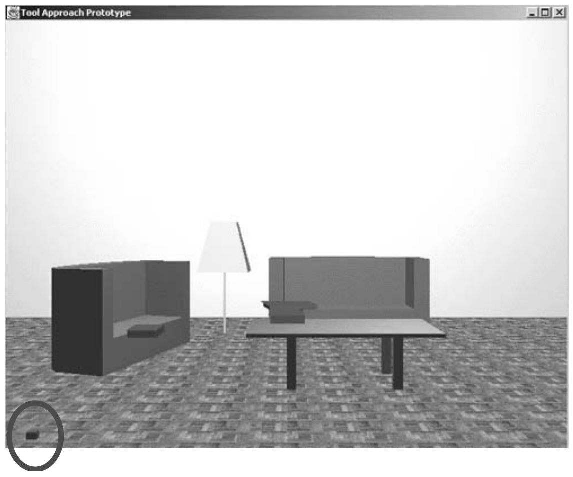


Figure 2 shows the toolbox, circled in the bottom left corner

While the toolbox is open, the tools appear for selection. These virtual tools are represented as buttons and arranged in a virtual menu, shown in Figure 3.

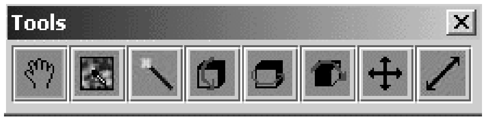


Figure 3 Shows the tools available once the box is opened

In Houde [11], the different shapes of the mouse cursor are used as an indication of the action performed. In our work, we follow this method – once the desired tool is selected, the shape of the mouse cursor will change accordingly. For instance, while the "paintbrush" is selected, the mouse cursor will change to a "brush" shape. While the toolbox is closed, the virtual menu disappears automatically and the shape of the mouse cursor changes to default (i.e. the arrow shape).

4.2 Pin System

As a comparison, we built an identical environment to that described above, the difference being that the interaction was conducted on a mode-per-object basis. This approach allows the users to "use" an object while the object is not in editing mode. For this reason, we have created objects to populate the scene, each of which has a default "use" function. For instance, the users can open and close the door or turn on the television set. As we mentioned earlier, we need a way to indicate the mode status of each object and an easy-to-understand metaphor to edit objects.

We have adapted our earlier idea of the screw indicating mode and have drawn our metaphor from the way that an artist draws a picture. In real life, an artist would put drawing paper on the drawing board and pin the paper on. The pin is used to fix the paper on the drawing board. It can be also seen as an indication that the drawing is in process. Therefore, to pin an object and then edit it is the metaphor we use in the virtual environment, eliminating the need to place a screw equivalent on every environment object.

As the metaphor required, a drawing pin is provided in the virtual environment. The drawing pin, similar to the toolbox, is placed at the left-bottom corner of the screen and it is always in this fixed location. The drawing pin will not block the users' view and it is always available despite of the users' position (the system re-places the pin in the object so that it is always visible). See Figure 4.

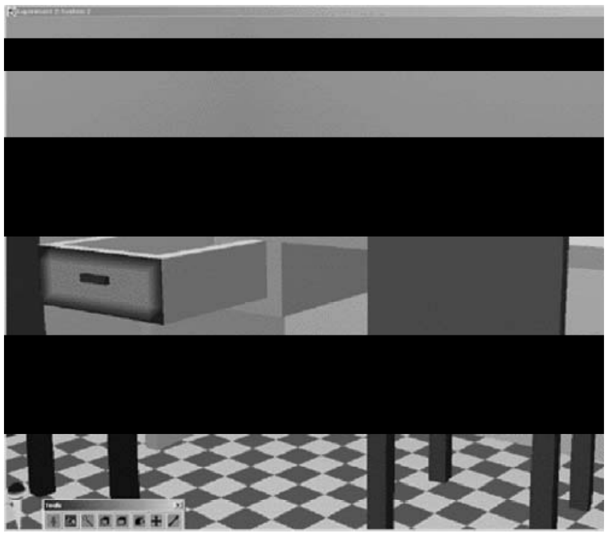


Figure 4. The drawing pin is at the corner of the screen (circled). The chair is pinned and the pin attached to the chair is in a different color from the drawing pin at the corner. While the chair is pinned, the users can still invoke the basic function of any unpinned object in the virtual environment.

To pin an object, the user simply drags the drawing pin and drops it on the object. To show that it is being edited, a new drawing pin will appear on that object. A tool list then appears from which the user can choose the desired tool and apply it to only that object. The users have to unpin the object in order to use the object, even when there is no tool mode set for that object. In order to avoid confusion between the drawing pin, which is always at the left-bottom corner of the screen, and the pin attached on object, we use different colors to differentiate class from instantiation.

We have provided visual feedback to indicate the currently active mode of the pinned objects. While the mouse is moving over the pinned object, the mouse cursor will change the shape according to the status of the object. Further feedback is provided via the texture on the pins, which are attached to the objects. The texture on the knob of the drawing pin will be the same as the active tools.

5. Study Design

For our study, we were able to find nine subjects who fitted the target user population we were interested in. These participants were paid volunteers and were students from various faculties in our university. We expect the end-users of these prototype systems to be non-experts in computer programming and computer graphics – they are interested in creating and editing their own virtual environments. However, they must be familiar with, and know how to use, standard computer input and output devices.

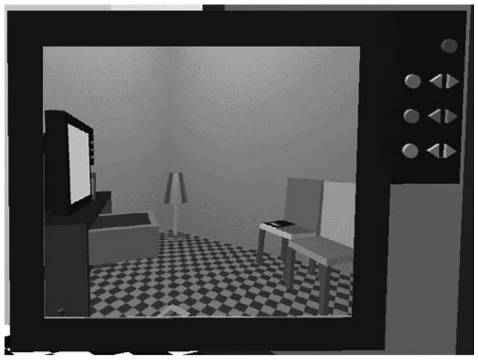
The experiment has a between-groups design. The participants were divided randomly into two groups and each was assigned to one of the prototypes. Four participants used the tool approach VE and five participants used the pin approach VE.

5.1 Tasks

There are two virtual rooms in the two prototypes. One room is a storage room, which contains all the furniture at the beginning of the experiment. The other room is a living room with a lamp inside. The users can walk freely in the virtual environment. However, they can only walk from storage room to living room through the door, and vice versa.

The task is to arrange the virtual living room according to the image in one of the virtual books. The users need to move all the furniture and objects to the other room (the living room), through the door. There are three books in the virtual environment. The books contain the images of three different arrangements of the room. There are three channels on the TV. On each channel, there are four images of the particular arrangements from different viewpoints. (One view is shown in Figure 5). The three books are marked differently at the back. One book is marked a “L”, one is marked a “1” and the last one does not have a mark. The users are asked to find the book with “1” at the back, and arrange the room accordingly.

We use the living room and storage room scenario as this is a real-world task familiar to users. Additionally, compared to manipulating some boxes, manipulating virtual furniture is more realistic. By placing instructions in books and on the television, we ensure that users are required to perform “use” actions in the environment – if users were give the plan on physical paper, they would not need to “use” the objects and hence comparison of the two interaction techniques would be pointless.



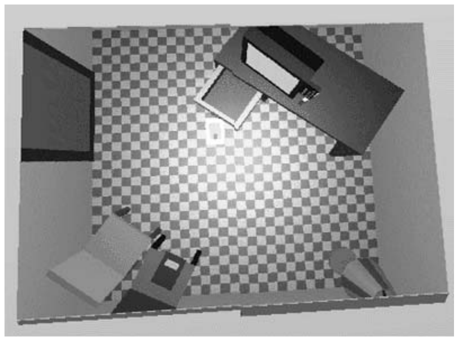


Figure 5. Pictures on the television set which shows users how to lay out the environment.

6. Measurement of Usability

We used two ways to measure the usability of both prototype systems, namely observation and questionnaires. The questionnaire we used is the Computer System Usability Questionnaire from AMC [12]. This was developed by J. R. Lewis of IBM and measures on a 7-point Likert scale. There are nineteen questions, which are based around three themes: system usefulness, information quality and interface quality.

Users were observed unobtrusively by splitting the monitor output lead and recording everything that happened on the user’s screen.

6.1 Tool Usage Observation Results

From post experiment video analysis, we have identified some behavioral patterns among the participants within each system and across both systems. With the tool approach prototype, we have found some patterns that the participants do while performing the task. These are summarized as following:

- Of all four participants, only one participant did not manage to complete the task. The other three managed to move all the furniture and books across the room and painted some objects.
- The users tried all tools on one object at the beginning.
- The users seemed to be familiar with the initially, but rapidly learnt how to use them efficiently.
- The participants moved objects to the other room in any order and put the objects in any position. Once all objects were in the living room, they then put the objects in the correct location. The last thing they did was to paint them.
- Some participants would put the book in the same orientation as the living room. That is, if they are standing in the storage and facing the living room, then they will place the book vertically and rotate it in such a way that the door is at the left bottom of the book.
- After using the system for a short while, the users understood that tools were applied to all objects. In other words, if they wanted to use the currently used-tool on other object, they did not have to click on the tool again.

6.2 Pin Usage Observation Results

With the pin approach prototype, we have identified some patterns of performing the tasks.

- Of all five participants, two of them did not complete the task of placing the objects correctly. The other three managed to move all the furniture and books across the room and only one of them did not start painting objects.
- When editing an object, no matter if the object was pinned or not, the participant would pin the object before starting.
- The participants would move objects to the correct position one by one. Once all objects are in the correct position, then they would paint the objects according the image in the book.
- Most participants did not fully understand the function of the drawing pin and would tend to invoke the functionality of the object while the object was pinned. Some participants only took a few mouse clicks to learn how to invoke the functionality of the objects; some took longer (about ten minutes). Eventually all participants learned before the end of the experiment. Once they learned it, they made fewer mistakes, and took fewer tries.

6.3 Overall Findings

There are some common behavioral patterns found in both systems.

- Even though the participants were informed that it was possible to walk through objects, they would try to not walk through objects.
- The participants would move objects away from each other if they collided.
- Most participants were confused with moving the object up and down (along Y-axis) and push and pull (along Z-axis). Even with exploring in the virtual environment for a while, some subjects were still confused with these two operations.
- To put objects one on top of the other, the users used all manipulation tools (rotation, movement, and push and pull) to make sure that the objects were lying flat on each other, even though this was not necessary (there was no physics modeling).
- The way of choosing color – using red, green and blue sliders – is not a good idea. Most subjects took a long time to find the desired colors.

6.4 Questionnaire Results

We asked the participants to answer the usability questionnaire (CSUQ). We present a summary findings.

5.4.1 Positive Comments

The subjects feel that using the mouse to navigate, and to interact in the virtual environment is easy to understand.

- Collision detection on the walls was good because it let them know if the object is against the walls.
- The manipulation method is intuitive, as if the users were in the real world.
- They feel that they have control of the environment.
- Negative Comments
- There is no Undo function.
- There is no Zoom function.
- Pieces of furniture can pass through one another and people can walk into objects.
- It is difficult to find the correct colors.

7. Discussion

From an analysis of the questionnaire results, it seems that the tool approach prototype is preferred to the pin approach prototype in terms of usability. This is also confirmed by observations, most probably because the users are more familiar with the tool interaction metaphor. Apart from that, the questionnaire picked up shortcomings in the prototype (e.g. no undo feature) rather than provide meaningful insight into the difference in interaction techniques.

We had expected that the users work “tool by tool” in the tool approach prototype and “object by object” in the pin approach prototype. However, there were no observed behavioral differences in working technique.

One interesting observation was that the participants who used the pin approach prototype would re-pin the objects, even if they could see the pin. It would seem that, rather than reflect on the state of an object, it was much faster just to pin it regardless. However, the participants who used the pin approach prototype more frequently invoked the function ("use" behavior) of objects, than those who used the tool approach prototype.

In both systems, the subjects avoid walking into objects and when one object collided into another, they moved one away from the other. This would indicate a high level of presence amongst the subjects, as they could have walked through objects put instead projected the attributes of the physical objects onto those in the environment. Many researchers suggest that high fidelity graphics are required to induce high levels of presence. However, our virtual environments are not photo-realistic and some physical laws, such as gravity, and collisions between users and objects are ignored. Nevertheless, the participants attempt to model the virtual environment as the real world, and obey the laws of reality without prompting.

8. Conclusions and Recommendations

Initially we thought that the pin approach prototype would be more useful than the tool approach prototype. One reason is that the ability to work on different objects with different modes is more efficient because users do not need to change modes constantly. Another advantage of the pin approach prototype over the tool approach prototype is that the pin approach is somehow more intuitive for the "use" functionality than the tool approach – the users do not need to use the "hand tool" to use an object in the pin approach.

That said, our observation of the users' actions shows that the users are more familiar with the tool approach than the pin approach. We suspect the reason it that global mode is more common in interactive software. As our subjects were all computer literate, this style of interaction may be more intuitive for them.

Although less suited to this task, it could be the case that the pin system would be more appropriate in an environment where the focus was primarily on use with only occasional editing required. This approach may also be suitable for collaborative virtual environments. The drawing pin can serve as a lock – while the drawing pin is on an object, others can see that the object is currently being edited. As the texture of the pin would change depending on the tools applied on that object, others in the environment can also know which operation is used on that object as well. These applications include architecture building, or design discussion meetings.

Acknowledgements

This work was made possible via NRF funding and Innovation Fund grants from the South African government.

References

[1] Alice Home Page http://www.alice.org/

[2] Dive Home Page http://www.sics.se/dive/

[3] Genisis Home Page http://www.genesis3d.com/

[4] Smith, R. B., Ungar, D. and Chang, B. The Use-Mention Perspective on Programming for the Interface. In Languages for Developing User Interfaces, Jones and Bartlett Publishers, Inc. (1992), 79-89.

[5] Poupyrev, I., Billinghurst, M., Weghorst, S., and Ichikawa, T. The Go-Go Interaction Techniques: Non-linear Mapping for Direct Manipulation in VR. Proc. UIST 1996, ACM Press (1996), 79-80.

[6] Stoakley, R., Conway, M. J. and Pausch, R. Virtual reality on a WIM: Interactive worlds in miniature. Proc. Human Factors in Computing Systems, (1995), 265-272.

[7] Heeter, C. Being There: The Subjective Experience of Presence. Presence: Teleoperators and Virtual Environments, (1992), 1(2): 262-271.

[8] Mine, M. R., Brooks, F.P. Jr. and Sequin, C. H. Moving Objects in Space: Exploiting Proprioception In Virtual-Environment Interaction. *Proc. SIGGRAPH*, ACM press, (1997), 19-26.

[9] Pierce, J. S., Stearns, B. C. and Pausch, R. Two Handed Manipulation of Voodoo Dolls in Virtual Environments. *Symposium on Interactive 3D Graphics*, pages 141-145, 1999.

[10] Marsden, G. Overcoming Design and Execute Modes in User Interface Design Environments. *HCI 95 people and Computers* (1995), 133-137.

[11] Houde, S. and Sellman, R. In Search of Design Principles for Programming Environments. Proc CHI'94, ACM Press (1994) 230.

[12] Perlman, G. Web-Based User Interface Evaluation With Questionnaires. http://www.acm.org/~perlman/

[13] Slater M., Brogni A., Steed A. Physiological Responses to Breaks in Presence: A Pilot Study. Proceedings of '6th International Workshop on Presence', Aalborg Denmark, 6-8 October, 2003.