Interfaces for Cloning in Immersive Virtual Environments

Jian Chen Doug A. Bowman John F. Lucas Chadwick A. Wingrave
Depart of Computer Science and Center for Human-Computer Interaction
Virginia Tech, Blacksburg, Virginia, USA

Abstract

Three-dimensional objects in many applications domains, such as architecture and construction, can be extremely complex and can consist of a large number of components. However, many of these complex objects also contain a great deal of repetition. Therefore, cloning techniques, which generate multiple spatially distributed copies of an object to form a repeated pattern, can be used to model these objects more efficiently. Such techniques are important and useful in desktop three-dimensional modeling systems, but we are not aware of any cloning techniques designed for immersive virtual environments (VEs). In this paper, we present an initial effort toward the design and development of such interfaces. We define the design space of the cloning task, and present five novel VE interfaces for cloning, then articulate the design rationale. We have also performed a usability study intended to elicit subjective responses with regard to affordances, feedback, attention, perceived usefulness, ease of use, and ease of learning in these interfaces. The study resulted in four major conclusions. First, slider widgets are better suited for discrete than for continuous numeric input. Second, the attentional requirements of the interface increase with increased degrees-of-freedom associated with widgets. Third, users prefer constrained widget movement, although more degrees-of-freedom allow more efficient parameter setting. Finally, appropriate feedback can reduce the cognitive load. The lessons we learned will influence our continuing design of cloning techniques, and these techniques will ultimately be applied to VE applications for design, construction, and prototyping.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Virtual Reality J.6 [Computer-Aided Engineering]: Computer-aided Design

1. Introduction

Lisa, a city planner, plans to build residential houses in a crowded city. She is planning the layout using immersive virtual environment (VE) techniques. She can walk through the model and move houses around using the current state-of-the-art VE interaction techniques. She wants to duplicate a house and put the copies in the virtual world just like she does when using desktop modeling tools, because a large amount of repetition exists. However, she can not perform such tasks in the VE. Rather, she has to go back the desktop environment, duplicate the house using a three-dimensional (3D) modeling tool, and reload the model into the virtual world.

Lisa's problem and others like it can be addressed through the using of cloning techniques designed for immersive VEs. Such techniques allow users to generate multiple spatially distributed copies of an object to form a repeated pattern, and therefore provide more efficient modeling of complex objects. Cloning is an example of a domain-specific 3D interaction task, since it takes the characteristics of the domain (architecture and construction in this case) into account. With a cloning technique, Lisa could automatically generate the houses she needs by defining the number of copies, the distance between adjacent houses, and the spatial location of the newly generated objects. Cloning techniques are important because they allow users to make complex modifications to the environment without leaving the virtual world. Usable cloning techniques can greatly reduce the time to build complicated structures in a VE..

The purpose of this research is therefore to define the design space of cloning techniques, to investigate how different interfaces reflect different design options, and to study the benefits of such interfaces in the construction and architecture domain. Our objective is to develop a complementary set of tools to make design changes possible in VEs, but not to substitute for existing 2D desktop tools or to make the user design structures from scratch.

We have developed five user interfaces for cloning, mainly classified by the types of the widgets employed and their degrees-of-freedom (DOFs). They are the numerical spinners interface (Spin), the orthogonal 1-DOF sliders interface (1-DOF-O), the cavalier 1-DOF sliders interface (1-DOF-C), the 2-DOF widgets interface (2-DOF), and the 3-DOF widgets interface (3-DOF). Each of these interfaces allows the user to quickly generate complex, repetitive models. For example, Figure 1 shows an example of a structure at a construction site and the 3D model of the same structure generated using one of our interfaces. It took less than 20 seconds to build the two-story building starting with only four beams and four columns.

Our research is novel in three respects. First, we allow users to build models directly within a three-dimensional (3D) environment with little effort. Second, we present multiple ways to handle numeric input, which is a difficult problem studied in the 3D user interface

literature [Min95]. Third, we present design rationale and compare the pros and cons of the interfaces in an exploratory study with regard to usability issues. The main contributions of our work lie in the systematic design method and empirical evaluation of cloning interfaces. The lessons learned allow other designers to avoid the problems we encountered, and also generalize to 3D interaction tasks beyond cloning.

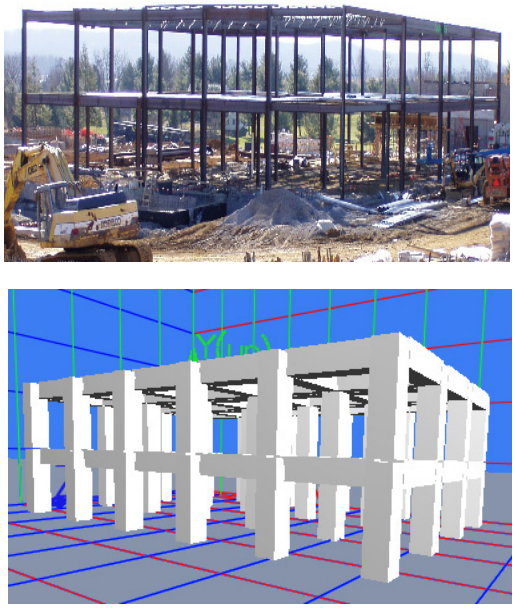


Figure 1: *Example structure at a real-world construction site and in our environment*

2. Related work

Recent years have seen a growing number of immersive design tools for computer-aided design. Sachs' 3-Draw [SRS91] and Deering's HoloSketch [Dee95] used a pen to draw shapes directly on a tablet for interactive 3D shape design. Butterworth's 3DM [BDH*92] developed a toolbox, an icon-based user interface, which was used to change the mode of operation for modeling a VE. Liang's JDCAD [LG93] presented many novel ideas, such as spotlight techniques for conic selection and ring menus for primitive creation, alignment, and reshaping using a 3D input device.

Bowman's Virtual Habitat environment [BWH*98] allowed user editing (resizing, reshaping, and positioning) of many types of objects, and constrained in the interaction using domain-specific information. Mine's ISAAC [Min97] used various menu display techniques and a world-in-miniature (WIM) to modify the space and create new elements. Most recently, Bowman's Virtual-SAP [BSP*03] allowed creating and manipulating architectural elements, such as beams, columns, and walls, to provide input to an earthquake simulation.

With these existing systems, however, it is still difficult to produce complex, but repetitive structures. A common characteristic of many of these systems was that they used direct manipulation to create or move vertices or objects. This had certain advantages, but also limited users' ability to model complex objects containing hundreds of elements or to provide precise positioning information. Our cloning interfaces, on the other hand, allow the user to generate complex structures quickly and with high levels of precision.

The most closely related techniques to ours are those within the 2D interfaces of 3D modeling tools such as Autodesk architectural desktop, AutoCAD, 3D Studio Max, and Risa. However, these techniques were designed for the desktop environment only and cannot be used in immersive VEs.

3. Parameters involved in cloning tasks

Cloning tasks have two major components: *selection* and *clone generation*. The user first *selects* objects of interest in the virtual world, then specifies the properties of the cloned objects. In our work, we have currently considered only the clone generation sub-task, and have defined a set of parameters for this sub-task describing how the newly generated objects are distributed in space and their visual attributes (Figure 2).

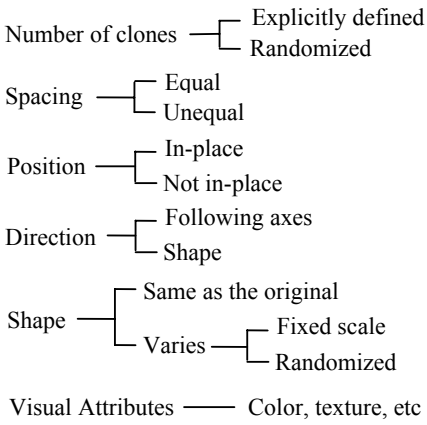


Figure 2: *Design space of cloning*

The *number of clones* can be a number explicitly defined by the user or randomly generated by the system. *Spacing* defines how far two adjacent copies are from each other, which could be equally or unequally distributed in the space. *Position* includes in-place, where the newly generated structure is part of the original structure, and not in-place, allowing the new structure to be put in any other position. *Direction* defines the area in which the clones are laid out, which may follow the axes of a coordinate system, or may follow a certain shape (e.g., a function defined by several points selected by the user). *Shape* and *visual attributes* deal with the look and feel of the objects.

Consider an example of using this design space to solve Lisa's problem presented in the introduction. Lisa can request 10x10 copies along the horizontal plane (e.g., along the x and z axes) and one copy along the vertical axis; she could then adjust the space between the houses to 25 feet.

This set of parameters for cloning focuses mainly on qualitative parameters and not on semantic constraints or rule-based attributes. For example, assume an environment with 10 tables, with a teapot on one of the tables, and suppose a user wants to put a teapot on each of the tables. Such tasks are not addressed in our parameter space.

4. Cloning user interfaces

The first step in the cloning process is to choose the parameters that we wish to control. We chose three parameters: number of clones, spacing, and direction. These parameters seem to be important because users can build rather complicated structures with reasonable flexibility using a single parameter or a combination. For the position parameter, we used in-place positioning where the original structure becomes a part of the newly generated structure.

Since all three parameters were conceptually different, multi-dimensional input strategies were required. The question was how to effectively specify the nine variables (three parameters in a 3D space) given different types of input: the three "number of clones" variables were discrete; the three spacing variables were continuous; and the three direction variables were binary in that they were either negative or positive.

We designed widgets to control the parameters. Widgets [CSH*92] are small objects with geometry and associated behavior. They have been widely used in 2D user interfaces. This allowed us to take advantage of existing users' familiarity and to allow the transfer of knowledge to the 3D interface. These widgets were displayed on a tablet, a tracked, hand-held palette (Figure 3). Users performed spatial input (such as pick, drop, or drag) using a tracked pen. This is called the pen-and-tablet metaphor [SRS91, AS95, SCP95, BSP*03].

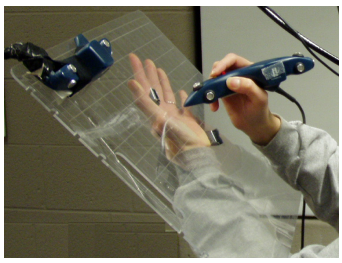


Figure 3: *Physical devices used in pen-and-tablet interaction metaphor*

4.1 Numerical spinners interface (Spin)

The *numerical spinners interface* (Spin) directly displays the nine variables on the tablet (Figure 4). Spinners were used to control the input of the number of clones and the spacing parameters; and radio buttons were used to change the direction, either on the positive or negative side of an axis. The display was organized in a clearly readable manner: three rows for the x, y, and z axes and three columns, for the parameters "counter" (for number of clones), "dir" (for direction), and "dist" (for spacing).

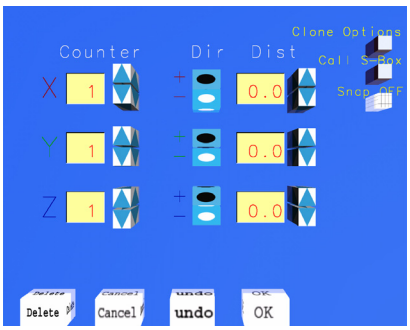


Figure 4: *Numerical spinners interface (Spin)*

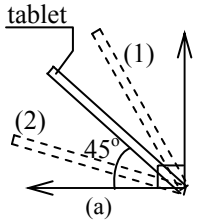
Users clicked the up and down arrows to increase or decrease the values. The corresponding variables were displayed in the text fields with a yellow background. The color black denoted the toggling radio buttons' current state. We also color-coded the text displays to match the axes drawn in the world, (red for x, green for y, and blue for z).

4.2 Orthogonal 1-DOF sliders (1-DOF-O)

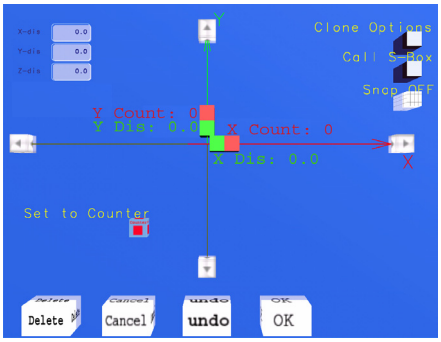
Slider widgets are often used in user interfaces to define variables. An example in 3D user interfaces is the work of Chen [Che88], who grouped the sliders together on the interface or attached them to objects within the world. Users manipulated the variables by directly controlling the widgets in 3D space.

We used six sliders to control the nine variables, two widgets on each axis to control the counter and distance variables. The interface was called *orthogonal 1-DOF-sliders* (1-DOF-O), because all sliders were constrained to move along one dimension (Figure 5). They were displayed in different colors: red-colored sliders controlled the number of clones and green-colored sliders controlled spacing. The direction variables were automatically accounted for by the spacing widgets since we allowed these widgets to indicate both positive and negative values. We constrained the movement of the

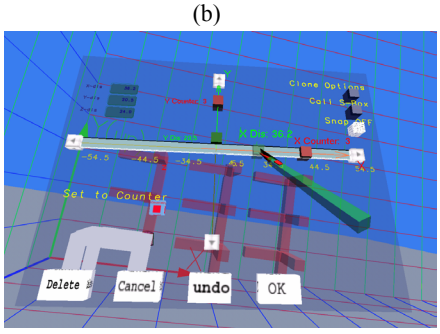
Example: The Y axis was visible if the tablet was located at position (1); and the z axis (point down) was visible if the tablet was at location (2).



(a)



(b)



(c)

Figure 5: *Orthogonal 1-DOF sliders interface (1-DOF-O)*

"number of clones" widgets to be along the positive axes because these variables are inherently positive.

To avoid clutter on the tablet, only four widgets were displayed at a time. The two widgets on the x axis were always visible. Either the two widgets on the y axis or on the z axis were visible depending on the angle between the tablet and the horizontal axis. If the angle was within a range of 0^o to 45^o (Figure 5(a)), the z axis and the two attached widgets were displayed; and if the angle was within a range of 45^o to 90^o, the y axis and its widgets were displayed. We chose to use angle to determine the mode because it is fast and easy for the users to rotate the tablet while performing the task. They only need to make a small adjustment of their hand or arm's position to switch axes. Such visibility constraints were made inactive when the user was interacting with a widget. This was because the user might get confused if the widget s/he was interacting with suddenly became invisible.

The interface also included *fine-adjustment widgets*. Clicking the arrow buttons shown at the end of each axis would move either the number of clones or spacing widgets along one dimension with predefined increments. Selection of which widget to move was controlled by another widget shown on the left bottom side of the tablet. "Set to counter" indicates that clicking on arrows would move counter widgets, and toggling it to "Set to distance" would cause the movement of spacing widgets on the tablet. The behavior of the arrow widgets was consistent on all interfaces when used.

The annotation and color themes used on the tablet were carefully designed for easy reading. The annotations were drawn on the screen (as a heads-up display). The current values of the variables were displayed next to the widgets in the same color as the widgets. The same color scheme was used to display the axes in the world. Also, the negative axis directions were drawn in a very different brown color for ease of interpretation.

4.3 Cavalier 1-DOF sliders interface (1-DOF-C)

Similar to the previous interface, in the cavalier 1-DOF sliders interface (1-DOF-C), all slider widgets were confined to move along one dimension. Instead of drawing the y and z axes along the same line on the tablet, we drew a 45^o cavalier projection in order to separate the y and z axes (Figure 6). All widgets were visible on the tablet.

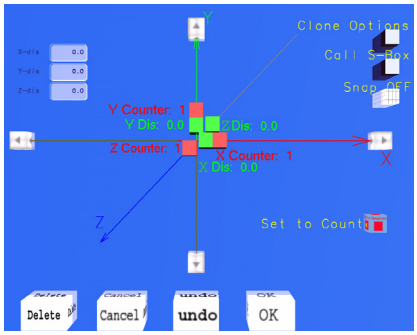


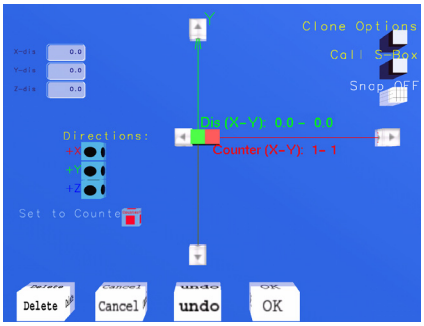
Figure 6: *Cavalier 1-DOF sliders interface (1-DOF-C)*

This projection visually gave the users a 3D feeling although it was a 2D user interface. We enhanced this effect by displaying the positive z axis larger than the negative z axis, which was visually further away. The behavior of all slider widgets was similar to those in the 1-DOF-O interface except that the z axis sliders were confined to move along the oblique z axis.

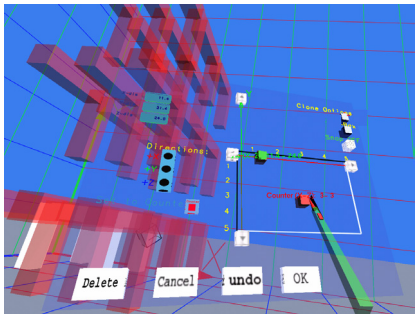
4.4 2-DOF widgets interface (2-DOF)

Widgets used in 2D desktop applications typically have few DOFs: they usually move along one dimension or along an axis. Sliders in the previous user interfaces were similar to their desktop counterpart. This is not a necessary constraint for widgets in 3D interfaces, where widgets can be moved within space.

We designed widgets whose movement was constrained to a plane, and therefore the interface was called the 2-DOF widgets interface (Figure 7). Dragging the widgets causes two variables along two directions to change simultaneously. For example, dragging the slider shown in Figure 7 (b) could change the number of copies in both the x and z directions. Similar to the 1-DOF-O interface, subsets of widgets were visible based on the angle of the tablet. Therefore, only two widgets were on the tablet at any given time.



(a)



(b)

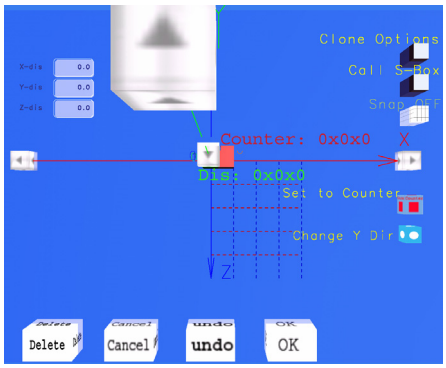
Figure 7: *2-DOF widgets interface (2-DOF)*

This interface provided for widgets' movement along the positive axes only. The direction was controlled by separate toggle widgets shown on the left side of the tablet. We chose this because the increased DOFs would make positioning tasks harder. In addition, the variables that were already set up were displayed next to the widgets.

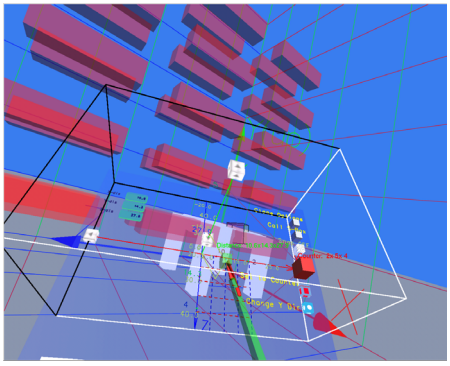
4.5 3-DOF widgets interface (3-DOF)

We increased the DOFs even more in the 3-DOF widgets interface, by allowing the widgets to be dragged in three dimensions within a box area defined by the size of the tablet (Figure 8). The x and z axes were located on the tablet, and the y axis pointed out of the tablet. Only two slider widgets were needed for this interface: one controlled three "number of clones" variables and the

other controlled three spacing variables (with directions). Because we wanted to put fine-control widgets at the end of each axis, the y-axis arrow widget ended up floating in the space above the tablet.



(a)



(b)

Figure 8: *3-DOF widgets interface (3-DOF)*

Since one objective of our study was to test how different design options affected usability, we combined the direction and the space variables again. A grid was drawn which looked similar to the one in the virtual world in order to assist the user to decide the direction. We added a toggle button marked "Change Y Dir" which allowed toggling the y direction on the tablet since the origin was located at the center of the tablet and the negative y direction was beneath the tablet, and therefore was difficult to reach.

5. Design Considerations

We considered cloning as an interactive process: users repeatedly changed the position of widgets, visually evaluated the outcome, re-adjusted the widgets, and repeated this process until the desired cloning result was obtained. Such a process created a fast feedback loop between the user input and result that significantly speeded up the process of interpreting the interfaces, making an action plan, and executing the tasks [Nor86].

We also tried to learn how different designs affected users' attention. We believed that a better interface would allow users to concentrate on their primary task rather than on the interface. Finally, continuous response and feedback were provided because of the constant interaction between the user and the application.

In the remainder of this section, we describe the rationale for several design choices that we applied to most or all of our cloning interfaces.

5.1 Self-explained constraints

We designed widgets to provide a perceived affordance for spatial inputs, to convey the behavior, and to inform the user how to act on them. We drew *slider slots* for 1-DOF sliders, boxes for 2-DOF widgets, and a cube for 3-DOF widgets upon selection (Figure 5(c), 7(b), 8(b)). We did not make them visible in the initial user interface due to limited space on the tablet and the overlap of the distance and counter axes. Visual clutter and overlap would make the interface hard to interpret, therefore increasing the cognitive load.

5.2 3D Preview

The common visual feedback for all of our cloning interfaces was a *3D preview*, i.e., previewing the result as the user interacts with the tablet. This provided a direct *what you see is what you get* (WYSIWYG) interface. The red-colored semitransparent objects displayed in Figures 5(c), 7(b), and 8(b) is the preview results corresponding to the user's current input. This output allowed users to see if their actions were furthering their goals.

The preview made generating the result an effortless and natural process, and may minimize the "consciously calculated activity" [BMB86]. Also, users do not need mental rotation [SM71] to visualize the results since the structure was displayed directly in the virtual world. This design feature bridges the gulf of evaluation (that is, it reduces the amount of effort required to interpret and understand the actions of the system) [Nor86], making results more easily interpretable.

5.3 Fine-control widgets

The fine-control widgets or arrow widgets allowed the movements of widgets to be constrained to a single dimension. 2-DOF or 3-DOF widgets might allow fast creation of large models, but the higher DOFs that users need to control may be frustrating because great agility and manual dexterity are required. Constraints, on the other hand, limit such movement to allow fast operation.

5.4 Appropriate annotation and visual feedback

Another source of feedback was semantic in nature [GJ91, BML*01]. Text was drawn on the tablet to indicate the current parameters in the system. The annotations around the widgets play two roles: (1) they help users choose the correct widget; and (2) they provide feedback for the values of variables. Furthermore, they were color-coded to augment the correspondence between the widgets and the text displayed.

5.5 Volume-based widgets

Fitt's law [Fit54, ZW03] suggests that the target acquisition is affected by the size of the target. We used both *volume-based widgets* (widgets with a magnetic area around them) and visual feedback to provide ease of selection. The volume-based widgets made the interface less cluttered and afforded easy selection. Also, widgets were highlighted upon touch.

5.6 Integration of users' domain knowledge

In the 1-DOF-C interface, a cube was drawn to represent the overall size of the cloned object within the space. This additional visual feedback might increase the understanding of the cloned structure in an abstract manner, and the users do not need to travel to get an

overview of the structure being cloned. We displayed this representation in this interface because of its natural structure and spatial relationship with the axes.

6. Usability evaluation

A good cloning technique is one that allows the users to accomplish tasks easily and efficiently with little or no discomfort. We performed an exploratory study to evaluate the usability of the cloning interfaces we had designed. The purpose of the evaluation was to (1) compare subjective usability responses aimed at investigating understandability, learnability, and ease of use, and (2) find usability problems from the perspectives of designers and other users regarding the content, aesthetics, and interaction techniques used in our system.

6.1 Participants

We recruited eight participants for this study. Two participants had construction domain knowledge and user interface design experience. The other six had user interface knowledge but no construction domain knowledge. They were all graduate students.

6.2 Equipment, environment, and software

The experiment used a Virtual Research V8 head-mounted display (HMD) with binocular display (640 x 480 resolution, 60^o diagonal field of view). The user's head, pen, and tablet were all tracked by an Intersense IS-900 VET tracker. The travel was pointing-based [Min95], where the orientation of the pen determines the direction of travel. Participants also used the pen to click (indicated by pressing a button), pick (indicated by pressing a button) and drag (indicated by holding a button) widgets.

The virtual world initially contained a single-story building with four beams and four columns. The size of the beams and columns were 5x5x20 units along x, y, and z directions, respectively. The counter widgets on the tablet had a maximum value of 5, and the distance widgets had a maximum value of 60 units. Clicking on the arrow button increased or decreased the selected counter widget's value by one unit or increased or decreased the selected spacing widget's value by five units. The prototype that integrated all the interfaces was implemented using the Simple Virtual Environment (SVE) library [KBH00] and OpenGL.

6.3 Tasks

Two tasks, a *matching* task and a *numeric* task, were used in the experiment. For the *matching task*, a miniature version of a structure was displayed near the tablet, and participants were asked to duplicate this structure in the environment.

For the *numeric input task*, participants were asked to assign specific values to the counter, distance and direction variables. An example task was "Generate a new structure that has three copies along the x axis, 4 copies along the y axis and 3 copies along the z axis. The distance between adjacent copies should be 10 units along the x axis, 30 units along the y axis, and 5 units along the z axis. The direction of the cloned objects should be along the directions of +x, -y, and –z."

The first task is closer to the real-world tasks. We used the second task because it still allowed us to find usability problems, while also requiring more precision. Due to the difficulty in setting up extremely precise numbers using spinners and widgets, we asked participants to complete the task within an error bound of +/-1 unit for each "spacing" variable. Tasks were read aloud for the participants and were also displayed on the bottom of the screen for their reference.

6.4 Procedure

Participants were asked to fill out a pre-questionnaire about their demographic information at the beginning and a post-questionnaire at the end of the study. The experiment was conducted in a quiet laboratory setting. We tested each interface with all participants, and counterbalanced the order of the conditions to avoid learning effects.

Three participants were not trained how to use the interfaces, and were asked to explore. Think-aloud protocol was used with these subjects. The other five participants were trained and were asked to complete the tasks as quickly as possible while avoiding errors. In this way, we evaluated the system from the perspective of learning and ease of use, and also from the perspective of productivity and efficiency.

We interviewed participants and discussed the interface issues after they completed the two tasks in each interface to avoid forgetfulness and confusion about particular interfaces. This also gave participants a short break, as the experiment lasted about two hours. Behavioral data, e.g., confusion, frustration, and user comfort were noted.

7. Results and discussion

7.1 Subjective usability ratings

We asked participants in the post-questionnaire about their preferences, perceived usefulness, and ease of use on a scale of 1 to 7, where 1 was the worst and 7 was the best. The perceived usefulness was defined as "the degree to which an individual believes that using a particular system would enhance his or her job performance;" and the perceived ease of use was defined as "the degree to which an individual believes that using a particular system would be free of physical and mental effort" [Dav89].

We can see from Figure 9 that for all of these measures, participants rated the Spin, 1-DOF-O, and 1-DOF-C interfaces highly. Participants' feeling of comfort with the interface got worse with increasing DOFs, which made 3-DOF the worst interface according to our subjects.

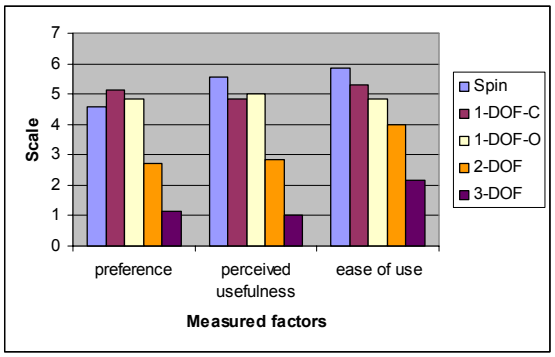


Figure 9: *Participant rating*

7.2 Interview results

Participants were asked questions on various aspects of the interfaces, such as: How easy was it to learn? Was it

easy to read the setup variables? Was the interface distracting? Was the interface cluttered? What did you feel about the feedback provided? Comments from both groups are listed below:

- Participants reported that once they learned one interface, it was very easy to use the others.
- Clicking on the arrows of the Spin interface was easier than dragging sliders on the other interfaces. But the Spin interface was not considered good for design because the spinners did not scale well; it could be impossible if an axis was required to scale into hundreds of units.
- The 1-DOF-O and 1-DOF-C interfaces were also reportedly very easy to understand and had a smaller incidence of errors. Three of the seven participants (including the two participants with construction-domain knowledge) did not like the disappearance of the third axis although they could get the invisible axis back very easily.
- The 2-DOF and 3-DOF interfaces were hard to use. Participants reported that it was very difficult to keep track of the position of the widgets for three variables. The maximum they could work on was two variables at a time. Most participants felt that using the 3-DOF interface was a distracting and stressful experience. But they also reported that they would use it if appropriate constraints were available and if less attention was needed.
- The feedback provided about the constrained moving area was enough for learning the behavior of 1-DOF sliders, but not of 2-DOF and 3-DOF widgets. Most participants did not comprehend the meaning of the rectangle on the 2-DOF interface and the box on the 3-DOF interface until they were told. Most participants did not have a problem with this feedback after being trained, but suggested that it needs to be more understandable.
- Most participants reported that the level of distraction increased with the DOFs of a widget instead of the number of widgets displayed on the tablet, and that increased DOFs also required more attention to the tablet.
- All three participants doing exploration reported that feedback offered in the 3-DOF interface was good, but they needed quite a bit of time to get used to it. However, most of the other five participants reported this interface was distracting, in that too much feedback was available on the tablet. All participants realized that they could build a structure very quickly using this interface, but fine adjustment was hard because of the higher number of DOFs available.
- Participants reported that controlling the "number of clones" widgets was easy, but controlling the spacing widget was difficult, due to the greater accuracy needed for the spacing.
- Most participants reported that they preferred to look at the parameters displayed next to the widgets compared to the scale marked on each axis on the tablet; and half of the participants did not use the scale on the axis at all.
- Participants suggested the separation of the spacing and the direction for 3-DOF interfaces due to the increased DOF. But they also said that the combination of spacing and the direction for the 1-DOF widgets was intuitive since they were fairly easy to move.

7.3 Observations

The main observation we made was that participants tended to set up one parameter at a time no matter how many parameters a widget could control. For example, when doing task 2 using the 3-DOF widgets interface, most participants dragged slider widgets along the x direction first, then along the y direction, then the z direction. The same behavior was observed while setting up the spacing variables. One participant with advanced 3D gaming experience tried to perform the task by setting two parameters at a time and then setting the third. He outperformed all other participants while using 3-DOF widgets on speed and accuracy.

7.4 Discussion

The spinner interface was easy to use because of its directness and the simple actions performed to change variables. It required less agility and manual dexterity from the user. The cognitive load of this interface was low because fewer items were displayed on the tablet during interaction. It was easy to use in that it was cognitively direct for numeric tasks: there were separate widgets corresponding to each of the nice parameters. Also, it provided sufficient accuracy for the tasks assigned to the participants. However, such alphanumeric input methods might not be efficient overall, as observed by Mine [Min95] and confirmed by our experiment. All of this is demonstrated by the high perceived usefulness and ease of use, but relatively low preference.

The constrained 1-DOF movement of sliders was a natural extension of the sliders used in desktop user interfaces, and therefore put less cognitive load on the users. Also, due to their flexibility in setting up the variables, participants preferred them to other interfaces. Although dragging the slider was a continuous action that could create a fuzzy range problem while being used to specify discrete numbers (such as the number of copies, which is an integer), the participants did not complain about this.

Free-space positioning was a difficult task which made higher DOF widgets harder to control. This made the 2-DOF and 3-DOF widgets the least preferred choice among all interfaces. The combination of direction and spacing control made participants frustrated when the participant finished setting up the distance then found the directions were not right. They then had to move the widgets to another coordinate which changed the spacing again. However, we feel that we should still work to improve the usability of these higher-DOF widgets because of their advantage in speed to create objects. This must of course be handled with more interaction constraints.

8. Conclusion and future work

Because complex but repetitive structures are common in the domains of construction and architecture, we developed five cloning interfaces that allow the generation of multiple distributed copies of objects while immersed in a virtual world.

We built a prototype to evaluate and compare our interfaces, and to find the important usability issues with regard to different design choices. The main take-away lessons from this study can be summarized as follows:

- 2D interaction with a pen-and-tablet in an immersive VE is qualitatively different than 2D interaction with a mouse on a desktop computer. It's very difficult for users to specify two or more DOFs simultaneously with a pen and tablet interface.
- Slider widgets in a 3D interface may be better suited for discrete than for continuous numeric input.
- The attentional requirements of the interface increase with increased widget DOFs. Provide separate parameter control to reduce the flexibility of the system with high DOFs.
- Constrain the widgets if possible.
- Provide appropriate feedback, reduce cognitive load and help users easily make action plans.

Our future work includes addressing the following questions:

- Can we provide effective numeric input interfaces to explicitly define parameters to satisfy the requirements of the construction domain?
- Can we improve usability by creating hybrids of these interfaces using the best parts from each?
- What other metaphors (such as direct manipulation or multi-modal input) might be appropriate for cloning? For example, users might directly manipulate the objects in the virtual world or a world-in-miniature. We will compare such techniques with the best techniques using our current metaphor.

This research will lead to detailed guidelines for specific aspects of the immersive design problem, and to the creation of general heuristics for domain-specific 3D interaction. This research can improve and push VEs into the everyday workflow of engineers and designers.

Acknowledgements

The work presented in this paper was sponsored by the National Science Foundation (NSF-IIS-0237412). The authors are grateful to the participants for volunteering for this study.

References:

[AS95] Augus, I, and Sowizral, H.: Embedding the 2D interaction metaphor in a real 3D virtual environment. *Proc. SPIE (Stereoscopic Displays and Virtual Reality Systems)*, 2409, (1995), 282-293.

[BDH*92] Butterworth, J., Davidson, A., Hench, S., and Olano, M.: 3DM: A three dimensional modeler using a head-mounted display. *ACM Symposium on Interactive 3D Graphics* (1992), 135-138.

[BMB86] Badler, N., Manoochehri, K., and Baraff, D.: Multi-dimensional input techniques and articulated figure positioning by multiple constraints. *ACM Workshop on Interactive 3D Graphics*, (1986), 151-170.

[BWH*98] Bowman, D.A., Wineman, J., Hodges, L., and Allison, D.: Designing animal habitats within an immersive VE. *IEEE Computer Graphics and Applications*, 18, 5, (1998), 9-13.

[BSP*03] Bowman, D.A., Setareh, M., Pinho, M.S., Ali, N., Kalita, A., Lee, Y., Lucas, J., Gracey, M., Kothapalli, M., Zhu, Q., Datey, A., and Tumati, P.: Virtual-SAP: an immersive tool for visualizing the response of building structure to environmental conditions. *Proceedings of the IEEE Virtual Reality*, (2003), 243-250.

[Che88] Chen, M.: A study in interactive 3D rotation using 2-d control devices. *Computer Graphics*, 22, 4, (1988), 121-129. (SIGGRAPH'88).

[CSH*92] Conner, D.B., Snibbe, S.S., Herndon, K.P., Robbins, D.C., Zeleznik, R.C., and van Dam, A.: Three-dimensional widgets. *Proceedings of the 1992 Symposium on Interactive 3D Graphics*, 26, (1992), 183-187.

[Dav89] Davis, F.D.: Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13, (1989), 319-340.

[Dee95] Deering, M.F.: HoloSketch: a virtual reality sketching/animation tool. *ACM Transaction on Computer-Human Interaction,* 2, 3 (1995), 220-238.

[Fit54] Fitts, P.M.: The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47, (1954), 381-391.

[GJ91] Green, M. and Jacob, R.: Software architectures and metaphors for non-WIMP user interface. *Computer Graphics*, 25, 3, (July 1991), 229-235. (SIGGRAPH'90 workshop).

[KBH00] Kessler, G., Bowman, D.A., Hodges, L.: The simple virtual environment library: an extensible framework for building VE applications. *Presence: Teleoperators and virtual environments*, 9, 2, (2000), 187-208.

[LG93] Liang, J. and Green, M.: Geometric modeling using six degrees of freedom input devices. *Proc. 3rd International Conference on CAD and Computer Graphics*, (August 1993), 217-222.

[Min95] Mine, M.R.: Virtual environment interaction techniques. *Technical Report, TR95-018*, Computer Science Department, University of North Carolina, 1995.

[Min97] Mine, M.: ISAAC: a meta-CAD system for virtual environments. *Computer-Aided Design*, 29, 8, (1997), 547-553.

[Nor86] Norman, D.A.: Cognitive engineering. In Norman, D.A. and Draper, S. (Eds), *User Centered Systems Design: New Perspectives on Human-Computer Interaction*. Erlbaum Associates, (1986), 31-61.

[SRS91] Sachs, E., Roberts, A., and Stoops, D.: 3-Draw: a tool for designing 3D shapes. *IEEE Computer Graphics and Applications*, 11, 6, (1991), 18-26.

[SCP95] Stoakley, R., Conway, M., and Pausch, R.: Virtual reality on a WIM: interactive worlds in miniature. *ACM SIGCHI'95*, 1995, 265-272.

[SM*71] Shepard, R.N. and Metzler, J.: mental rotation of three dimensional objects. *Science*, 171, (1971), 701-703.

[ZW03] Zhai, S., Woltjer, R.: Human movement performance in relation to path constraint - the law of steering in locomotion. *Proceedings of IEEE Virtual Reality*, (March 2003), 149-156.