Axis-based Multitouch Interfaces for 3D Manipulation

Abstract

We present intuitive multitouch interfaces for interactive 3D object manipulation. These interfaces are widgetless and buttonless, allowing the user to directly manipulate 3D objects without operating complex widgets or performing mode switching or tool selection. Exploiting a real-time finger registration method and context-sensitive operation analysis, the interfaces directly map different multitouch gestures to different 3D manipulation tasks. User interaction is greatly simplified by using a single touch gesture that simultaneously specifies the transformation type, orientation, and applies the transformation to the focus object. In addition, active snapping and axis transfer functionalities are supported. The former allows the user to snap together two objects by simply drawing a free touch path connecting them. The latter allows an object to be transformed along a predefined axis of another object. Our pilot evaluation indicates that users can efficiently use our interfaces to construct simple 3D scenes after only a short training period.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Interaction Techniques; H.5.2 [Information Interfaces and Presentation]: User Interfaces—

1. Introduction

Multitouch techniques are now commonly used in various applications. Many digital devices, from large desktop computers to handheld mobile internet devices, are now equipped with touchscreens or touchpads supporting multitouch operations. However, the use of multitouch input in real-world applications have so far been restricted to enhancing navigation and browsing functionalities, for example, browsing of images and maps on a virtual plane or navigating 3D spaces using multitouch supported rotation and zooming operations like in Google Earth [GOO].

3D modeling has made significant impact in the engineering, design and entertainment industries. However, the interfaces provided in commercial modeling systems are mostly designed for use with keyboard and single-point-input (mouse or pen devices). Traditional 3D modeling software are notoriously difficult to use due to their complex user interfaces that include a large set of mode-switching buttons and keyboard shortkeys. 3D manipulation is particularly frequently used and are therefore assigned the most accessible buttons and keys. Ideally, the use of buttons and keys should be avoided in touch-based systems.

While the traditional 3D manipulation widgets found in commercial modeling systems can be directly integrated into touch-based interfaces, the design of these widgets is based on a tool-switching metaphor which conflicts with the tool-free philosophy of multitouch environment. Further, small widgets in standard interfaces are difficult to operate due to the relatively larger finger-tip blob.

There has been little research on using multitouch input for complex 3D editing such as manipulation of multiple objects in 3D space. In fact, multitouch input contains rich orientation and transformation information, allowing user to provide multiple input data with only a single multitouch action. This avoids tedious editing steps such as mode/tool switching and item selection required in traditional modeling environments. In this paper we present a novel multitouch interface for direct 3D object manipulation that is widgetless and buttonless, and supports imprecise touch-based input without sacrificing control power and usability.

Our system supports most manipulating capabilities found in commercial 3D modeling interfaces. In addition to standard translation, rotation and scaling operations, our system supports snapping of two arbitrarily located objects and manipulation relative to an arbitrary reference frame. The convenient transfer of reference frames between objects is enabled by our axis transfer interface, which further simplifies relative transformation tasks among multiple objects.
Figure 1: Examples of models created using our multitouch interface.

Figure 1 shows two examples which created by our proposed interfaces within a few of minutes.

2. Related Work

Since the early days of development of computer graphics, researchers have been investigating manipulation of 3D objects in interactive environment. Early solutions used indirect controls such as sliders to control the transformation of objects [ETW81]. With the increased computation power, real-time systems based on basic 3D widgets like skitter and jake became possible [Bie87].

In general, 3D widgets are any visual elements for manipulating objects in the virtual environment. Earlier research focused on general frameworks for 3D widget design [CSH92, SZH94]. Later, interactive systems for specific applications were developed [GP95] [CSB05] [Han97]. Although widget implementations vary from applications to applications, the most common manipulation tools consist of three-axis widgets that allow axis-based translation, rotation and scaling, which are uniformly supported by commercial 3D modeling software [AUTa,AUTb,BLE].

While 3D manipulation widgets are now extensively used, they are in fact mainly designed for use with keyboard and single pointing devices (e.g. mouse or pen). There are other systems developed for use with two pointing devices bimanually for 3D manipulation [ZFS97,BK99], however they have not been popularized to the general market.

To support the increasingly popular sketch modeling paradigm, Schmidt el al. [SSB08] presented an interface for 3D object manipulation using transient 3D widgets defined by sketched context-dependent strokes. The widgets are automatically aligned to the object’s axes determined by the user’s stroke. It provides widget interaction for coarse object positioning and orientating. Our interface design is inspired by their approach and shares the same goal of minimizing interface complexity and user interaction. We choose a widgetless approach that exploits the rich information of multitouch input, leading to a seamless interface that allows user to use single touch actions to specify the transformation axis, transformation type, and at the same time apply the transformation.

More recently, due to the rapid development of multitouch display devices [DL01, Han05], researchers have been exploring multitouch interfaces for virtual object manipulation. Direct manipulation techniques for 2D free-form translation, rotation and scaling were first introduced [KCST05, HCV06, SVFR04]. Then several extensions from multitouch 2D manipulation to 3D virtual spaces have been proposed, such as the works of Hancock et al. [HCC07, HCC09] and Reisman et al. [RDH09]. These works focus on mapping direct touch inputs to 6DOF control (location and orientation) of 3D objects in the virtual space. These techniques provide intuitive control, especially freeform rotation, for 3D objects. This is highly suitable for casual browsing applications that provide easy interaction for user to browse and locate 3D objects. However, due to the lack of precise axis-based manipulation, they are not suitable for modeling and composition tasks. In addition, these techniques mainly focus on single object manipulation and global operations such as changing of view-point, they do not support multiple object manipulations such as alignment and object duplication. In contrast, our proposed interface enables precise axis-based transformations and integrates with global view-point changing, objects snapping, focusing on efficient 3D composition as provided in traditional 3D applications.

Recently, Au and Tai [AT10] presented a simple finger registration technique that can distinguish in real-time which hand and fingers of the user are touching the touchscreen. Their method is able to distinguish different hands and fingers directly from the positions of the contact points whenever the user places a hand in any orientation anywhere on the touchscreen. We make use of this registration technique to design our interface for 3D object manipulation. It allows the design of rich multitouch interactions that associate different operations to touch actions involving different palm and finger combinations, resulting in seamless interactive manipulation without explicit mode and tool switching.

3. Design Overview

In general, any interactive editing process involves three main kinds of operations – regions/objects focus selection, tool selection and manipulation. Take for example editing a document with word processing software, the user first needs to select the correct page and location to be edited, s/he edits the content using a tool, such as highlighting the selected text by pressing a toolbar button. These three operations (focus selection, tool selection and manipulation) consume the majority of the interaction time, therefore convenient and efficient switching between operation modes and manipulation tools is the crucial part of intuitive interface design.

Interface design for 3D modeling is even more challeng-
Figure 2: 3D transformation widgets used in Autodesk Maya. (Left) Universal Manipulator. (Right) Move / rotate / scale tool.

To enable direct and seamless manipulation, we chose to discard all widgets in our user interface design. Our general goal is to design a "widgetless" 3D modeling environment powered by multitouch input. As multitouch inputs contain rich orientation and transformation (motion paths of touch points) information, they suit the needs of 3D editing applications which require multi-dimension inputs. Incorporating the recently proposed finger and palm registration method [AT10], we define rich and easy-to-use gestures to replace the traditional complex widgets of non-multitouch systems.

User indicates an object to manipulate by explicitly selecting it with a single tap. After that, user can apply a touch action (single touch or multitouch) to manipulate the selected object. Rather than relying on explicit selection of the operation mode and editing tool by widgets and buttons, we adopt a context-sensitive operation analysis to determine the user-desired operation to be performed based on the current selected object, the user’s action and the registered palm and fingers. This allows designing an interface that directly maps different multitouch gestures to different types of 3D manipulations. User interaction can be greatly simplified by designing single touch actions that specify the transformation type, the orientation, and at the same time apply the transformation to the focus object. Our interface supports basic manipulation tasks including the global browsing operations which control the camera or change the viewpoint and the axis-based manipulations which transform objects to new positions and orientations.

For most manipulation tasks, we choose to use the motion paths and orientation information of the touching points, rather than their contact locations to determine user’s desired operations. Users are not required to precisely touch any specific objects or UI elements to specify their intentions (recall that no popup widgets are used). In any single manipulating operation, our interface constrains the object along a specific predefined axis, this decreases the overall interface complexity. However, this does not reduce the degrees of freedom of editing, as user can always define or reuse the axis set of different objects using the axis transfer operation. With the non-locational input and constrained axis manipulation, our interface is tolerant to inaccurate input and provides a casual editing style which is suitable for non-professional users.

Although our multitouch interface avoids mode and tool switching to change viewpoint and object transformations, in general it still requires multiple interaction steps to transform an object to a desired new position in the 3D virtual space. For simple stacking and snapping manipulations, our system provides the active snapping operation that allows user to directly snap two 3D objects by drawing a free touch path connecting the snapping objects. This avoids tedious operations required with the standard transformation manipulations. In addition, our interface supports axis transfer operation that allows relative manipulation (e.g. rotate one object along an axis of another object) to be accomplished easily. We design a simple two-hand interface for the axis transfer operations, with one hand selecting the reference object (the axes provider) and the other hand applying a basic manipulation operation along a specific axis of the reference object, similar to a normal editing scenario.

Figure 3: Automatic finger registration. The palm and fingers are registered based on the relative positions and orientations of the finger tips [AT10].

4. Finger and Palm Registration

To enable richer control with multitouch input, our system makes use of a recently proposed automatic and real-time finger and palm registration method to determine which palms and fingers of a user are touching the screen during editing [AT10]. Whenever the system detects multiple
contact points within a short interval, the finger registration procedure is invoked to determine which hand of the user is touching the screen and which contact point belongs to which finger. The registration process is based on the relative positions and orientations of the finger tips when three or more fingers are touching the screen in a natural pose. The process is performed in real-time and it supports placing of fingers in any arbitrary location and orientation (Figure 3 left).

Since the finger registration is only activated whenever three or more contact points are detected, it can be concurrently used with other touch-based interfaces that do not require finger registration, such as scaling with two fingers (see Section 5.3 axis-based manipulation) or manipulation with one finger (e.g. active snapping in Section 5.4).

5. Multitouch Interface Design

A key characteristic of our interface design is enabling seamless browsing in 3D space and 3D manipulation. We adopt a context-sensitive approach to determine the user’s desired operation based on the current selected object(s), the registered palm and fingers, and the motion and orientation of the contact points.

5.1. Global Browsing

Global rotation is one of the most frequently used manipulations in interactive 3D applications, since from time to time users need to check the current editing scene from different viewing directions. Therefore we choose to use a simple touch path (with one contact point) for the global rotation operation. Our UI adopts the virtual trackball interface [CMS88, HSH04] as the rotation controlling tool. Objects and scene can be easily rotated about any axis with the virtual trackball interface. In our design, the start and end points of a touch-and-drag gesture specify the orientation and magnitude of the rotation (see first row of Figure 4). User activates the trackball interface by touching the screen using one finger, dragging that finger to the desired end position and finally raising the finger to complete the operation. Smooth transformations are achieved while the end point is moved by the user, providing instant and continuous feedback. Note that we do not support global rotation about a specific axis (e.g. the world x-, y- and z-axis), since precise viewing direction is usually not needed for interactive manipulation. In contrast, we provide axis-based transformation operations for precise manipulation of individual objects, but do not support free-form transformations of individual objects.

For global panning (panning the view point/camera parallel to the screen), our UI uses the right palm translation (with 5 fingers touching the screen in a natural pose) to activate the operation. It considers the screen as a large paper on the table and the panning operation simulates shifting of the paper (see second row Figure 4). In order to avoid accidental activation of global panning, we rely on the palm and finger registration method in [AT10] to check if all the contact points are from the same palm.

For global zooming (moving the view point/camera forward and backward), our UI uses the fingers radial movement of the right palm to activate the operation. It simulates the pulling and pushing actions corresponding to the zoom in and zoom out operations (see last two rows of Figure 4). Similarly, we use the finger registration to check whether the contact points are from the same palm.

5.2. Object Selection

User needs to select the target object(s) in the scene before applying any subsequent manipulation. Individual object is selected by tapping (i.e. a touch-and-raise operation) the object using one finger. The selected object is highlighted in red, indicating that it is the focus object to receive the subsequent operation (Figure 5 left). Multiple object selection
Figure 5: Object selection. (left) A 1-finger tap to select an object. (right) A 5-finger tap to include the new object under index finger into current selection.

Figure 6: A double tap to select an object and center it at the same time.

is also possible. To extend the current selection, user uses a 5-finger tap with the new object to be included in the selection under the index finger. The newly selected object then changes to blue, indicating that it is now included in the current selection (see Figure 5 right), while the focus object remains red. We will present operations performed on multiple objects in Sections 5.4 and 5.5.

Centering Object. To facilitate easy manipulation of the focus object, we provide a fast way for the user to center it on the screen. The user can combine selection and centering using a double-tap. That is, in addition to the object being selected, the viewing point is panned automatically such that the selected object is centered on the screen (Figure 6).

5.3. Axis-based Manipulation

We envision basic manipulation of 3D object achievable with two-point gestures, since such gestures suffice for specifying 2D position, orientation and transformation information in the screen space for editing purposes.

Since a 2D orientation in screen space cannot uniquely define a 3D axis or orientation, we predefine a set of candidate axes for the focus object (or the focus group of objects). The candidate axes usually form an orthogonal frame but are not necessarily so. For simplicity, we define them as the face normals of the boundary box of the selected objects. Other possibilities are using the face normals of the convex hull of the objects or segmenting surface to obtain the patch orientations. User uses a 2-point touch to specify a 2D orientation which is then compared with the projected directions of the candidate axes. The candidate axis with orientation most similar to the specified 2D orientation is used for the current manipulation (Figure 7). From viewpoints where an axis is nearly parallel to the viewing direction, we disallow selecting that axis since transformations along that axis are unstable due to a short distance in screen space corresponding to a large distance in 3D space.

Note that the axis selection process is activated by a 2-point touch only when there is a focus object. After an axis is selected, user can immediately apply manipulations, giving a seamless and smooth manipulation integrating object selection, axis selection and editing into one single action.

Recall that our system is widgetless, not requiring user to touch specific positions on the object or elements of widgets to perform editing. Only the orientation and motion given by the contact points are considered. This reduces the difficulty of editing using multitouch input since no complex widgets need to be handled and no precise tapping and selection are required.

Axis Translation. With the axis of an object selected, if the user moves the touching fingers along the axis direction (i.e., translate the contact points parallel to the axis orientation), we designate that gesture as axis translation. The system translates the object according to the moved distance of the contact points (Figure 8 top). The amount of translation is according to the displacement of the contact points on the screen, allowing the whole screen space to be the input region and thus more easily achieve precise control.

Axis Rotation. Orienting 3D objects using a 2D input device is difficult for non-experienced users. This is one of the reasons why 3D modeling systems have steep learning curves. The virtual trackball technique can provide intuitive control, however it is difficult to specify precise orientation. Users are significantly faster and more accurate when performing simple single-axis rotation [CMS88]. All commercial 3D modeling systems provide constrained rotation widgets to support single-axis rotations (see Figure 2). In our UI design, axis rotation is activated when the two touching fingers are moved perpendicularly to the selected axis (Figure 8 middle).

Axis Scaling. Like most 3D modeling systems, our UI supports two different types of scaling manipulations. The
first type is uniform scaling, which scales simultaneously in all three dimensions. Similar to global zooming, we design to use the radial movement of the left hand’s fingers to activate this operation, except that now the object to be scaled has been selected. The second type is axis scaling. User contracts or expands the distance between two contact points along the selected axis (Figure 8 bottom) to scale the focus object along the chosen axis. The amount of scaling (enlarge or shrink) is defined by the initial distance between the two contact points and their displacements.

Buffer Zone. The user-desired manipulation operations are determined based on the relative movement of the contact points, namely, 2D translation (for axis translation and axis rotation) and 2D scaling along a line (for axis scaling). For robustness and avoidance of mis-interpreted operations due to unstable user touch input, our system ignores those movements of touch points within a given short distance. The transformation type of contact points and user desired operations are determined only when the contact points have displacements larger than the given threshold. This approach also allows the user to cancel the current manipulation by moving his/her fingers to the initial touch positions, causing the contact points to be located within the buffer zone and thus no manipulation is applied. Note that this buffer zone idea is similar to the magnitude filtering technique used for 2D manipulation in [NBBW09]. The difference is that for simplicity of control we only allow one of the operations (translation / rotation / scaling) to be applied in one user action, while [NBBW09] allows users to apply combination of these operations in a single action.

5.4. Object Snapping

Snapping is a very effective way of specifying accurate transformations. Interactive snapping techniques are extensively used in 3D modeling systems, especially engineering CAD systems, as they avoid tedious menu or dialog operations. We also provide automatic snapping in our system to reduce user’s interaction. We predefine the snapping planes and edges for each input object, which are defined by the surface normals of the object bounding box. In the case of complex objects, the snapping planes can be found by analyzing the surface normals of the object.

The snapping planes and edges define the snapping conditions during manipulation. The object being manipulated is snapped to another object if a pair of their snapping planes or edges are close to each other and have similar orientations (Figure 9 top). In the cases when two or more possible snapping pairs are detected, the snapping pairs are ordered according to their distances and the snapping pairs that conflict with previously applied ones are ignored (Figure 9 bottom).

Active Snapping. Automatic snapping is simple to use, however it still requires the user to place the object at an approximate location and orientation to activate the snapping. It is useful and efficient to allow user to directly snap or stack objects together without first transforming the selected object. We call such a functionality active snapping, which requires only a single action to specify the snapping objects and how they are to be snapped. User draws a free

Figure 8: Axis-based manipulation. (from top to bottom) Translation: move two contact points along the chosen axis; Rotation: move two contact points perpendicular to the chosen axis; Scaling: Contract or expand the distance between two contact points to scale object along the chosen axis.

Figure 9: (top) Automatic snapping between two snapping planes with similar orientations. (bottom) When multiple possibilities are detected, snapping pairs are ordered according to their distances and conflicting pairs are discarded.
touch path connecting two objects to be snapped. By projecting the normals of the snapping planes of the focus object to the screen space, user can easily select a snapping plane using the drawn path. Specifically, the snapping plane of the object with normal parallel to the starting orientation of the path drawn by the user’s finger is selected (Figure ??). Similarly, the target object and its snapping plane are selected according to the ending location and orientation of the path (see two examples in Figure 10). Since we are using the beginning and ending orientations of the touch path to specify the snapping, user can snap objects even when the snapping faces are not visible to the user. This avoids additional rotation of the view or objects needed by other interfaces that require user to specify the points for snapping, like [SSB08].

Overall, this operation avoids the otherwise tedious movement and rotation of the objects, giving a convenient interface for efficient multiple object editing.

5.5. Axis Transfer

A common limitation of most 3D editing systems is that, while it is easy to select the canonical axes of the editing object, setting up arbitrary axes is often a difficult task. It requires setting up a pivot or frame object, which incurs additional manipulation. However, often the user’s desired axis is simply a candidate axis of another object. For example, in Figure 11, to duplicate the leg of the toy dog model, the selected object group needs to be translated in the coordinate system of the body part. Thus what we need is transferring of axes between objects. Our solution to this task is a two-hand gesture, with one hand selecting the reference object that provides the new axes and the other hand manipulating the target object with respect to a specific axis of the reference object (Figure 11 right). This solution naturally enables the user to easily construct new canonical axes for editing via placing a simple reference object as proxy.

5.6. Other Supporting Operations

Object Duplication. Object duplication is an important basic operation in many 3D editing scenarios. Our multitouch modeling system supports object duplication in two ways. We call the first way active duplication, which is similar to axis translation, except that instead of translating the selected object, here a new copy of the selected object is created then translated (Figure 12 top). Active duplication is activated with a 3-finger translation (with thumb, index and middle). The translation direction of the fingers decides the translation axis.

The second way is called transformed duplication, which requires two selected objects. It creates a new copy of the focus object and transforms the new copy of focus object to a new location based on the transformation between the two selected objects. This operation is supported by many commercial modeling systems and is sometime referred to as “advance duplication”. Our system provides this operation to facilitate construction of more complex scenes (Figure 12 bottom). Note that the selected objects need not be of the same type since only the relative transformation between the objects is needed for the duplication.

Atomic Operations. There are other supporting operations that do not require any orientation and positional information but are necessary for a complete interactive editing system. We call them the atomic operations as they are usually simply activated by selecting a toolbar button or menu item in traditional editing systems. Examples are the undo/redo operations, and loading new objects or removing selected objects from the current scene. To integrate these operations into our multitouch system, we use the Palm Menu in [AT10] to provide a menu system that enables fast access and minimizes finger and eye movement. The menu is activated by a 5-finger left-hand tap and the popup buttons are placed exactly under the finger contact points (Fig-
Figure 12: Object duplication. (Top) Active duplication with 3-finger translation: a new copy of selected object is created then translated. (bottom) Transformed duplication: the focus object (red) is duplicated and then translated based on the transformation among objects in the selected group (red and blue objects).

Figure 13: A five-finger tap activates the Palm Menu, which has menu buttons located exactly at the finger touch points.

Figure 14: Participants were asked to construct the castle from given basic blocks using our multitouch interface.

6. Evaluation and Discussion

We conducted pilot experiments to investigate the effectiveness of our interface design. Six graduate students from computer science department were involved in these informal tests. All participants were experienced computer users and about half of them regularly use multitouch applications such as image and map browsing tools on mobile devices. None of them has extensive experience with graphics modeling software. Since one of our objectives is to provide intuitive multitouch interfaces for non-professional users, we did not directly compare our system to commercial 3D modeling software, nor other traditional single-pointing-based and widget-based interfaces. We record the feedback about the controllability of our interface and the participants’ acceptability of the interface.

Participants were asked to perform an assembly task of constructing a castle with different shapes of blocks (Figure 14). There were no accuracy requirements; participants were free to use the proposed manipulation gestures in any order to construct a castle as similar to the one shown as possible. We then collected feedback from each participant after he/she completed the session.

The results of these pilot experiments indicate that our interface design is effective and has good controllability for basic 3D manipulation. Each participant takes different time spans to familiarize with the manipulation operations, depending on their experience with 3D manipulation, from several minutes to half an hour. All participants can successfully construct the castle within 10 to 20 minutes. Participants who are regular users of multitouch mobile devices take relatively less time to learn our interface and complete the assigned task. All the participants reported that after the learning session, they can easily place the objects at the desired locations. The active snapping operation is particularly convenient for placing object at precise locations. Note that, with only standard transformations, due to the lack of parallax in 2D display, user would need to change their viewing angles to examine whether the objects are placed at the specified locations. With the active snapping operation that allows automatic alignment of objects in a single view, users can avoid such examinations and adjustments, leading to fewer view-angle changings.

Some participants have experience playing 3D games and using 3D browsing applications like Google Earth [GOO]. Testing results show that they have better understanding of the basic 3D orientation concepts, such as view point, view angle, transformation axis and transformation type, and they are able to better control 3D objects. In comparison, other
participants who lack the understanding of visualization of 3D space on 2D display reported frustration initially in controlling 3D objects that float in the virtual space. They need to spend more time familiarizing with the basic 3D orientation concepts, starting from changing view angle with virtual trackball to selecting suitable axis and transformation operation.

Participants reported two minor issues after completing the task. They cannot rotate objects with a rotation axis nearly parallel to the viewing direction (recalls that we disallow the selection of such an axis due to unstable control and difficulty in determining the projected axis orientation). Although using a global rotation to change the viewing angle can handle this issue, it introduces extra steps in the interactive editing process. Some participants suggest extending the current interface to support screen-space rotation, for both the camera (global rotation) and the focus object (local manipulation).

The second issue is that participants sometime forget to first select an object before applying the active snapping operation (thus they actually perform a global rotation with the free-form path). This is the limitation of the context-sensitive interfaces since users need to remember which operations are supported under the current selection context. However, we found that once the dependence of the context-sensitive interactions were explained, users rarely make such mistakes. In contrast, we noticed that users do not have this problem with the axis-based transformations, apparently because the candidate axes of an object are displayed only after the object is selected. This extra visualization information gives an implicit hint to user about the operation order. Note that for the active snapping, there is no such auxiliary visualization to indicate that snapping can only be applied to the focus object.

7. Conclusions

In this paper we proposed a novel multitouch interface for interactive manipulation of 3D objects. It provides widgetless translation, rotation and scaling, passive and active snapping, and axis transfer functionalities. Multi-touch input contains orientation and transformation information which is naturally suitable for 3D object manipulation. Utilizing these information, we designed a widgetless and context-aware interface that allows user to directly manipulate objects in the virtual space, based only on multitouch input, without any mode switching or tool selection as with traditional interfaces.

We implemented and evaluated our approach. Our exploratory study shows that non-professional users are able to manipulate 3D objects using our interface to build simple scenes after only a short training period. The active snapping technique provides a simple way to stack and align 3D objects without requiring multiple transformations and repeated changing of view direction, thus further reduces user interaction. With the axis transfer operation, users have the ability and freedom to define arbitrary canonical axes, allowing object transformation according to any orientation.

There are a few possible directions for future study. One is to enrich the visualization and improve the interface design in order to provide more hints for context-sensitive operations, thus reduces the occasions of incorrect operations. It is also possible to integrate more kinds of manipulations into our interface design, for example, supporting screen / world space transformation for individual objects, or allowing objects to be moved along curves or surfaces, rather than only along the predefined axes. Note that our current interface design mainly focuses on single-hand operations, there are rooms for designing more complex operations using both-hand manipulations. Finally, as a pilot study, our results are positive and encouraging. However, a more comprehensive and formal evaluation is still needed for future development.

References


[Han05] HAN J. Y.: Low-cost multi-touch sensing through frustrated total internal reflection. In UBI ’05 (2005), pp. 115–118.


