TouchSketch: a touch-based interface for 3D object manipulation and editing

Siju Wu, Amine Chellali, Samir Otmane, Guillaume Moreau

TouchSketch: a touch-based interface for 3D object manipulation and editing

Siju Wu
IBISC, Evry University
Evry, France

Amine Chellali
IBISC, Evry University
Evry, France

Samir Otmane
IBISC, Evry University
Evry, France

Guillaume Moreau
CERMA, Ecole Centrale de Nantes
Nantes, France

Abstract

To make constrained manipulation of 3D objects in desktop 3D applications, 3D transformation widgets are commonly used. However, their performance degrades on touchscreens because of low accuracy of touch inputs and the fingertip occlusion problem. In this paper, we present TouchSketch, a touch-based interface which allows users to perform independently fine-grained object translation, rotation and scaling on mobile devices. Our manipulation technique permits using the non-dominant hand to specify the manipulation reference while the dominant hand is used to determine the operation mode and control the transformation. In addition to 3D manipulation, TouchSketch provides also a set of functions to edit the shape of 3D objects. Users can use this application to accomplish rapid sketching tasks. We have conducted a user study to evaluate the efficiency of our manipulation technique. The results show that our technique outperforms a Widget-based technique regarding both the efficiency and fluency.

CR Categories: H.5.2 [Information interfaces and presentation]: User Interfaces—Graphical user interfaces (GUI), Interaction styles; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques

Keywords: Touch input; interaction techniques; 3D object manipulation.

1 Introduction

In many commercial desktop 3D modeling applications, standard 3D transformation widgets are commonly used to make fine-grained manipulation of objects. Transformation widgets are well designed to take advantages of the high accuracy of mouse inputs. In addition to mouse inputs, keyboard shortcuts are commonly used to allow the users changing the manipulation modes more efficiently. Because of the prosperous development of Smartphones and Tablet-PCs, new requirements of performing 3D sketches and manipulating 3D models on mobile devices have emerged in the recent years. However, through observing users interacting with standard 3D transformation widgets on touchscreens, Cohé et al. have found that the performance of standard transformation widgets on touchscreen suffers from a relatively low accuracy of touch inputs and the fingertip occlusion problem [Cohé et al. 2011].Thus, it is difficult to specify the desired DOF using a standard manipulator on touchscreens. It was also observed in this work, that users are disturbed by ergonomic issues during operations. To facilitate touch-based manipulation, 123D Design developed by Autodesk, provides a group of redefined transformation widgets with a larger size. However, one drawback of this solution is that users have to click buttons frequently to change the operation mode. As a result, both the efficiency and consistency of the design work are degraded.

Manipulation of 3D objects consists of three main tasks: rotation, scaling and translation (RST). To allow users constructing precisely 3D objects and scenes, it is common to provide the possibility of specifying simultaneously the manipulation task and the manipulation constraint. Because touch inputs are more intuitive but less precise than mouse inputs, it is necessary to propose new manipulation techniques which support imprecise touch inputs without sacrificing their usability. Previous studies such as Eden [Kin et al. 2011], tBox [Cohé et al. 2011] and Toucheo [Hachet et al. 2011] provide different solutions to perform RST in all three dimensions. One limitation of these techniques is that manipulation gestures should be performed upon the target object or a proxy widget. Performance of these techniques may be affected by the object cluster density. In addition, some interaction gestures proposed by these techniques are difficult to perform on mobile devices because they require the use of both hands. As the non-dominant hand (NDH) is often used to hold the device, its motor space is very limited. In this paper, we propose a new manipulation technique for touch-based paradigm based on the bimanual asymmetrical model proposed by Guiard [Guiard 1987]. Our technique allows users to specify the manipulation reference using the NDH while the dominant hand (DH) is used to determine the manipulation mode and control the transformation. One advantage of this technique is that axis- and plane-constraints can be specified simply by using the NDH without changing the holding posture. Moreover, object manipulation can be made without touching the target. Therefore, the performance is less affected by the object position, orientation and scale.

Besides proposing a new object manipulation technique for touch-based devices, we have also designed a GUI which provides a group of functions to edit the shape of 3D objects. For example, users can transform or extrude a face to modify the shape of an object. For these functions which are commonly implemented in desktop sketching applications, we have redefined their interaction methods to adapt to the touch-based paradigm.

In the following, we first discuss the state-of-the-art of touch-based manipulation techniques. Then, we present the design of our manipulation techniques and the sketching interface. Finally, we present a user study that aims to compare our technique with two other state-of-the-art techniques.

2 Related Work

2.1 Manipulation techniques

In 3D applications, manipulation of 3D objects is one of the most fundamental and essential functions. Existing solutions for object manipulation can be divided into 2 groups according to the mechanism of 3D manipulation: unconstrained manipulation and constrained manipulation. Unconstrained manipulation allows users to control 3D objects more freely while constrained manipulation enables users to specify an axis- or plane-constraint to manipulate

objects in a more precise way. In the category of unconstrained manipulation, several works have been done to control virtual objects in the same way as a physical object. Previous work in [Wilson et al. 2008] [Wilson 2009] and [Cao et al. 2008] have tried similar strategies to simulate physics on the surface. Users can manipulate virtual objects through plausible gestures such as grasping and pushing. The system can detect the gesture being performed and simulate forces on objects with the help of a physics engine. Although these interfaces provide a vivid way to manipulate virtual objects arbitrarily, they can only be used for coarse manipulations. Several works have proposed solutions for semi-constrained manipulation. Hancock et al. have proposed a way to control object rotation coupled with translation by using only one, two or three touches [Hancock et al. 2007]. This solution is intuitive; however translation and rotation cannot be performed separately when necessary. Reisman et al. have proposed a formulation for 2D and 3D direct manipulation [Reisman et al. 2009]. By analyzing touch movements on the screen, the system can infer the objective of users and manipulate the object in the desired way. The operation mode and transformation range depend on the count of touch inputs, contact regions on the object and movements of constraint points. Because the different DOF are not separated clearly, this technique is not appropriate for fine-grained manipulation as well.

For constrained manipulation, a few touch-based solutions exist. Schmidt et al. have proposed a set of sketching and composing widgets for 3D manipulation [Schmidt et al. 2008]. This technique allows users to draw different strokes to summon various manipulation widgets upon the selected object and perform RST with the help of manipulation widgets. Although this technique supports manipulation in all 9 DOF, learning all the gestures is difficult for novice users. Eden is an interface developed to construct animation scenes [Kin et al. 2011]. A set of gestures are provided to translate, rotate and scale objects independently. Although it permits to accurately manipulate 3D virtual objects, one drawback of this technique is that users have to keep pressing one finger on the manipulated object during manipulation. Unlike Eden, tBox [Cohé et al. 2011] and Toucheo [Hachet et al. 2011] are two widget-based manipulation techniques. tBox displays a box-shape widget which covers the selected object and users can perform RST by interacting with edges and faces of the box. The setup of Toucheo displays floating objects above the touchscreen and a circle-shape widget is displayed right beneath the selected object. A set of simple gestures are proposed to manipulate objects by controlling the widget. The advantage of these techniques is that manipulation is less affected by the size and orientation of target objects. Mendes has proposed a solution which is called LTouchIt to manipulate LEDO bricks on multi-touch surfaces [Mendes 2011]. Users can perform a pick gesture to specify a brick and translate it directly in a vertical or horizontal plane by dragging the hand. The translation plane can be switched by a secondary touch input. LEGO bricks can be also rotated by controlling a rotation handle with the help of both hands.

Liu et al. have proposed a set of two-finger gestures for 6 DOF manipulation [Liu et al. 2012]. This technique allows using only the dominant hand to translate and rotate 3D objects in three directions. Au et al. have also designed a set of manipulation gestures by leveraging two fingers of the DH [Au et al. 2012]. The advantage of this technique is that manipulation reference, operation mode and control of transformation can all be performed by only one gesture. After setting the manipulation constraint using the direction of two touches, users can trigger the desired manipulation by performing the corresponding gesture in a seamless way. In addition, manipulation gestures can be performed anywhere in the empty space. However, in some situations, the DH should be held in an uncomfortable way to set the manipulation direction. Moreover, if projections of two axes overlap each other, it is not easy to select the right one. Based on this work, we propose a new technique which allows specifying manipulation constraints without restriction of the camera viewpoint. Our technique allows using both hands simultaneously to manipulate objects without changing the holding posture of the NDH hand. Contrast to aforementioned techniques such as tBox and LTouchIt, our technique does not require users to perform gestures in the proximity of objects to trigger manipulation, therefore it is more suitable for interaction on TabletPCs.

2.2 Sketching application on mobile devices

Few studies have been conducted to design new interfaces for 3D modeling on touchscreens. Mockup Builder demonstrates how 3D sketching can be done on and above the surface [De Araújo et al. 2013]. Users can first draw a 2D primitive on the screen and then extrude it to the third dimension using gestures in air. Sun et al. have proposed a touch-based interface for fast architectural sketching [Sun et al. 2013]. A set of natural and intuitive gestures are designed for drawing the building outline. Similarly, Paper3D presents a powerful touch-based interface for paper folding work [Paczkowski et al. 2014]. It is possible to manipulate the virtual paper in a very familiar way to construct complex 3D models. In this paper, we propose a new 3D sketching application based also on a group of common-used 3D modeling functions to help designers edit 3D objects. The interaction methods of common-used 3D modeling functions in our application such as face manipulation and face extrusion are redesigned to adapt to the touch-based paradigm.

3 Manipulation Technique

3.1 Design rationale

In this paper, our main objective is to redefine 3D sketching interfaces on mobile devices for designers familiar with traditional desktop software. Following an iterative design process, we first conducted a task and needs analysis through discussion with 3 expert users of 3D modeling software. This permitted to identify several requirements that should be met by the touch-based interface. First, users should be able to manipulate an object without keeping fingers pressed upon it. In fact, it is suited to avoid misoperations caused by the occlusion issues, more particularly when multiple objects are close to each other. Second, 3D objects should be controlled in all 3 dimensions independently. It is also necessary to permit switching between the world coordinate system and the object local system. Therefore, users can specify the appropriate coordinate system according to manipulation tasks. Third, due to the lack of keyboard shortcuts, a substitution mechanism should be provided to switch between transformation modes seamlessly. Fourth, because the camera is frequently manipulated during design and sketching tasks, the proposed technique should work properly under any camera viewpoint. Among all techniques discussed in our literature review, the work of [Au et al. 2012] satisfies the first three requirements. However, the performance of this technique is sometimes degraded by the camera viewpoint. For this reason, we propose a new technique that aims to meet all four requirements.

When manipulating 3D objects in desktop modeling applications, intermediates and experts use the DH to control the transformation widget while the NDH is used to switch operation modes by pressing keyboard shortcuts. The coordination of two hands can be described by the bimanual asymmetrical model proposed by Guiard [Guiard 1987]. Guiard has shown that during daily tasks, the DH is used to perform fine movements and to manipulate tools, while the NDH is used to set the spatial frame of reference and to issue coarse movements. Subsequently, we decided to explore how this

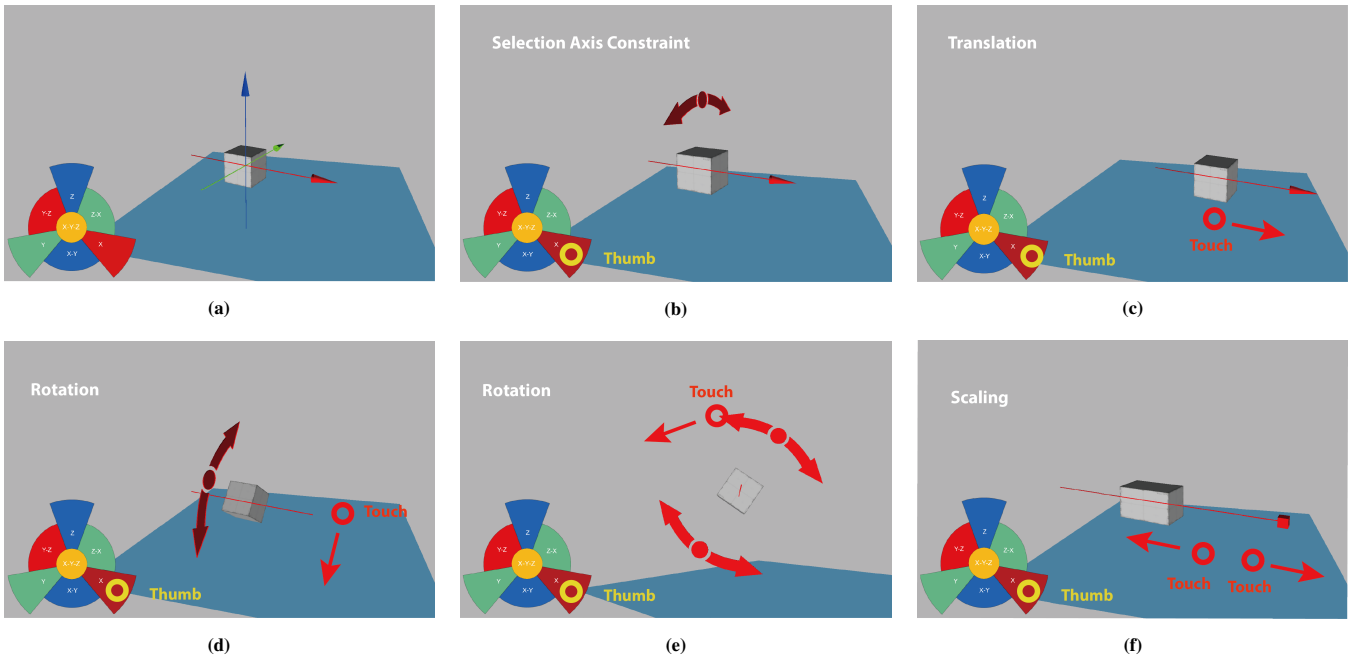


Figure 1: *Gestures for axis-based transformation manipulations. (a) The initial state of an object before manipulation. (b) After specifying the X-axis constraint, only the red axis is displayed. (c-f) After an axis-constraint is selected, users can translate, rotate and scale the object by using DH.*

model can be adapted for use on mobile devices to facilitate object manipulations.

However, because the NDH is usually used to hold the device and only the thumb can be used to make inputs, bimanual interaction techniques can be difficult to use on mobile devices. To increase the expressiveness of the thumb, some previous studies such as MicroRolls [Roudaut et al. 2009] and ThumbRock [Bonnet et al. 2013] have proposed to map different thumb movements on the screen to various operations. Boring et al. have used the size of the contact area to switch functions [Boring et al. 2012]. Wagner et al. have designed bimanual interaction on tablets in another way [Wagner et al. 2012]. They have first made a user study to identify all the common postures that are used to hold tablets. Then, they have designed a set of interactive widgets which are located inside the motor space of fingers of the NDH. Their controlled experiment demonstrated that these interactive widgets could help improving interactions. Thus, we decomposed the manipulation task into two subtasks and assigned each of them to a different hand. Because the NDH is commonly used to set the spatial frame of reference for the DH, we decided to use it to specify manipulation constraints. After a constraint is specified, the DH can then be used to trigger the operation and control the transformation through only one gesture. We will first present how a constraint can be specified by the NDH, and then explain how RST manipulation can be performed by the DH.

3.2 Constraint Menu

When performing fine-grained manipulations, 3D objects are always controlled with a constraint. Depending on the type of the constraint, manipulation can be divided into three categories: axis-constrained manipulation, plane-constrained manipulation and uniform manipulation. To facilitate the selection of constraints through the use of the NDH, we have designed the Constraint Menu.

As shown in Figure 2, the Constraint Menu is a marking menu with seven options which allows users to select one option among three axis-constraints, three plane-constraints and the uniform mode. Three fan-shaped options of larger sizes which are labeled 'X', 'Y' and 'Z' respectively, are used to set axis-constraints, while the other three smaller fan-shaped options which are labeled "X-Y", "Y-Z" and "Z-X" respectively, are used to set plane-constraints. Finally, the circular button in the center is used to trigger the uniform mode. For right handed users, the Constraint Menu is displayed in the left down corner of the display (Figure 1) to make it reachable by the thumb of the NDH. The interface configuration can be inversed for left handed users. Because axis-constrained manipulation is performed more frequently than plane-constrained manipulation, we decided to display sectors of axis-constraints with larger sizes than those of plane-constraints to highlight them. Moreover, the shape difference between two kinds of constraint sectors also makes them more distinguishable. Because X-axis, Y-axis and Z-axis are perpendicular to YZ-plane, ZX-plane and XY-plane respectively, each pair of axis and plane is displayed in the diagonal configuration and they also share the same color. It is important to note that objects manipulation can only be made when the thumb is pressing the corresponding constraint on the Constraint Menu.

3.3 Translation, Rotation & Scaling

Once an axis-constraint is specified, the selected object can be translated, rotated or scaled according to the selected axis. As shown in Figure 1 (b), when the X-axis constraint is selected, only the red axis is displayed. Object translation can be performed by dragging one finger of the DH along the selected axis (Figure 1 (c)). Rotating the object also requires dragging one finger, but in the perpendicular direction of the selected axis (Figure 1 (d)). Both the amount of translation and the amount of rotation are calculated according to the average distance between the current contact point and the initial contact location. Our translation and rotation techniques are similar to those of [Au et al. 2012], but can be performed

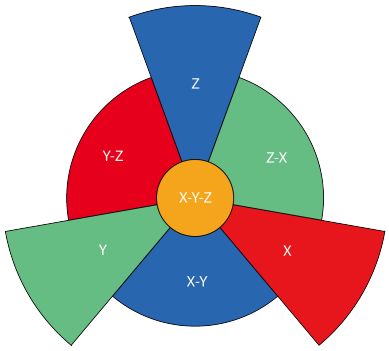


Figure 2: *The Constraint Menu which is used to specify manipulation constraints.*

by only one finger. During a pilot study, we found that it is difficult to rotate the object when the axis is almost perpendicular to the screen. In this case, the projection of the axis on the screen is too short to determine in which direction the finger should be panned to trigger rotation. To overcome this issue, we have designed another rotation method based on the rotation algorithm applied by tBox [Cohé et al. 2011]. When the constraint axis is perpendicular to the screen, two interactive rotation arrows are displayed (Figure 1 (e)). Users can tap one arrow and rotate it around the object to control the rotation. When the drag direction can be easily determined, only one arrow is displayed to provide visual hint but it is not interactive. Unlike object translation and rotation, object scaling can be performed by using a pinch gesture. Moving two fingers apart (or towards each other) can scale up (or down) the object (Figure 1 (f)). The pinch gesture can be performed in any direction. In other words, it is not necessary to perform the pinch gesture in the direction of the axis.

When a plane-constraint is specified, users can manipulate the object in a similar way. However, only translation and scaling are supported. As shown in Figure 3 (a), a semi-transparent plane is displayed to indicate the selected plane-constraint. Users can drag freely one finger on the screen to translate the selected object along the plane or perform the pinch gesture to perform uniform scaling in the plane (no scaling along plane normal).

When the circular button is pressed to set the uniform manipulation, performing the pinch gesture can launch uniform scaling (Figure3 (b)). Because in design and sketching tasks it is rare to translate objects freely in all three directions simultaneously, our technique does not support object translation when the uniform mode is activated.

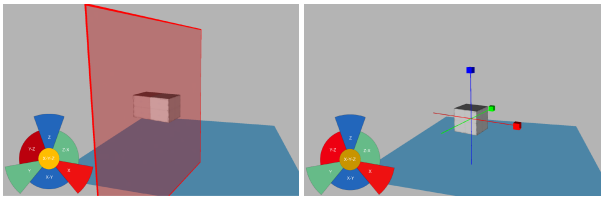
In general, our technique supports the following features:

Position, orientation and scale independence: Except object rotation around an axis perpendicular to the screen, manipulation tasks are not affected by the object position, orientation and scale. Manipulation gestures can be performed anywhere on the screen using the DH.

Coordination of both hands. The NDH and the DH can be leveraged together to accomplish object manipulation. The NDH can be used to specify the manipulation constraint without changing the holding posture.

Low memory load: Classic interaction gestures such as panning and pinching are reused for manipulation. This facilitates the memorization of gestures and thus the learnability of the technique.

Unaffected by the camera viewpoint: Our technique allows users



(a) (b)

Figure 3: *(a) When a plane-constraint is selected, the corresponding plane is displayed. Users can translate and rotate the object along the plane. (b)When the uniform mode is activated, the object can be scaled uniformly.*

to manipulate objects under any camera perspective. Because manipulation constraints can simply be selected from the Constraint Menu, object manipulation is less affected by the camera position and orientation.

4 TouchSketch

Besides proposing a set of manipulation techniques, we have also developed a user interface which is called TouchSketch to enable objects shape editing on tablets. Through discussion with expert users of desktop sketching software, we have identified a set of functions which are commonly used during design and sketching work. To adapt our interface to the touch-based paradigm, we have redefined the interaction methods of these functions to improve the user experience on touch-based mobile devices.

4.1 Primitive generation

As shown in Figure 4, to start sketching, users can first press one button in the right side to select a primitive to be generated. Then, they can tap the screen to generate the selected primitive in the touched location. A grid is displayed in the scene as a spatial reference and the bottom of each new generated primitive is attached to it.

4.2 Object & Face Selection

An unselected object can be selected by tapping it. Tapping it a second time cancels the selection. It is also possible to select multiple objects. To deselect all the selected objects, the user needs to tap the finger in the empty space. As shown in Figure 4, when the finger is pressed on the object without being released, two circular buttons are displayed. The finger can be dragged to the left button to select the face of the touched object when the finger is dropped. Dragging the finger to the right button deletes the touched object. After selecting an object or a face, the Constraint Menu is displayed.

4.3 Object duplication

After discussing with expert users, we found that duplicates of objects are commonly generated and then translated along one principal axis. To meet this requirement, we have designed a copy gesture that performs object duplication and translation of the copy at the same time. To copy a selected object, users specify first one axis-constraint. After that, they drag three fingers of the DH along the selected axis (Figure 5). Once the dragging distance passes a threshold value, the object is copied and its duplicate can be translated along the axis. If a plane-constraint is selected, the copy of the object can be translated along the selected plane.

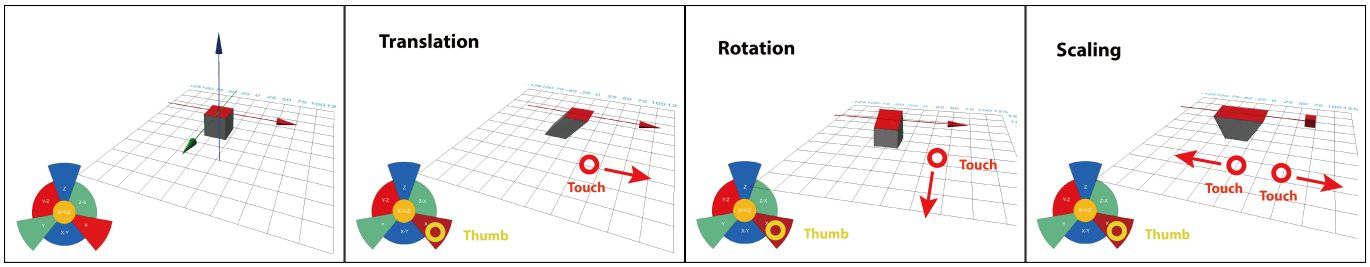


Figure 5: *(a) A face is selected. (b-d) The selected face can be translated, rotated and scaled in the same way as an object.*

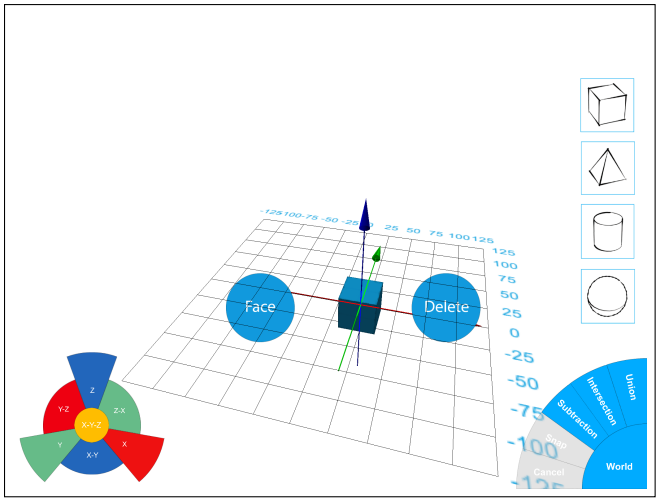


Figure 4: *The interface of TouchSketch.*

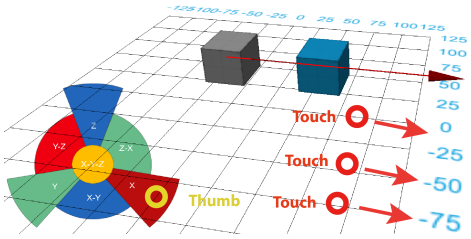


Figure 6: *After an axis-constraint is specified, dragging three fingers on the screen and copy the selected object and translate its copy along the specified axis.*

4.4 Face manipulation

After a face is selected, it can be manipulated like an object. As shown in Figure 6, object manipulation gestures can be reproduced to translate, rotate and scale the selected face. It is important to note that face manipulation is made in the local coordinate system of the selected face. In other words, the x-axis and the y-axis of the system are respectively parallel to the tangent and bi-tangent of the selected face. The z-axis is parallel to the normal of the face. The face center is set as the origin of the coordinate system. Since the height of a planar face is to 0, performing the scaling gesture on the z-axis has no effect on the object shape. Reusing object manipulation techniques to control selected faces reduces the learning cost and requires users to retain less information.

4.5 Face extrusion

Face extrusion is a common function in commercial sketching applications that permits to extrude one face from its initial position to modify the object shape. Because face extrusion is a very practical function when modifying the shape of objects, we have proposed a method to perform this operation in a fluent way. Face extrusion can be performed in two ways. First, after a face is selected, users can directly tap three fingers of the DH to extrude the face without specifying a constraint. In this case, the original face and the extruded face overlap each other. This operation is equivalent to duplicating the face in its initial position. In addition, users can also extrude the face along an axis-constraint. After an axis-constraint is specified, and similar to object duplication, dragging three fingers can extrude the face along the selected axis (Figure 7). The extrusion range can be controlled by the dragging distance of fingers. Through combing the face duplication and control of extrusion range, the process of face extrusion is simplified.

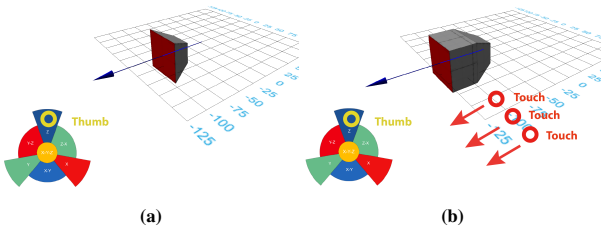


Figure 7: *(a) Object before extrusion. (b) The shape of the object is modified after one face is extruded.*

4.6 Camera Navigation

The scene of the sketching environment is displayed using a perspective camera. When no manipulation constraint is selected, the camera can be manipulated. Dragging one finger on the screen can orbit the camera on a virtual trackball. Performing the drag and pinch gesture with two fingers can pan and zoom the camera respectively.

4.7 Function marking menu

When an object or a face is selected, a marking menu is also displayed in the right down corner of the interface (Figure 4). Clicking the center of the menu permits to switch between the world coordinate system and the object local system. The selected object can

be manipulated with respect to the active coordinate system. Constrained translation, rotation and uniform scaling can be performed in both the world and local coordinate systems, while axis- and plane-constrained scaling can only be performed in the local coordinate system.

Besides switching the manipulation reference, the marking menu provides additional functions. For instance, when an object is selected, users can choose a boolean operation from the menu and select a target object to calculate the intersection, union or subtraction between them. If the subtraction operation is selected, the intersection of two objects is subtracted from the first selected object. If a face is selected, users can choose the snapping operation and select another face of the target object. Then the first object is transformed to stick its selected face to the touched face of the target object.

5 Evaluation of the manipulation technique

To evaluate the performance of our manipulation technique, we have conducted a controlled experiment. The objective was to determine whether the Constraint Menu technique outperforms other state-of-the-art techniques regarding its efficiency. For comparison, we have chosen the manipulation technique of 123D Design and the technique proposed by [Au et al. 2012]. Different from standard transformation widgets, the manipulation widgets of 123D Design have been redefined for utilization on mobile devices. However, its performance has never been evaluated. The technique of 123D Design is reproduced as shown in Figure 8. Users can press the translation, rotation and scaling buttons in the left side to activate the corresponding transformation widgets. The technique proposed in [Au et al. 2012] allows users to achieve manipulation on 9 DOF by using only the DH. Moreover, it outperforms the standard transformation widgets. Therefore, we decided to compare the performance of these two techniques to ours. In the following, we use Widget-Based technique to refer to the technique of 123D Design and Duo-Finger Constraint technique to refer to the technique of [Au et al. 2012].

Although some other techniques such as Eden [Kin et al. 2011] and tBox [Hachet et al. 2011] can be used to make fine-grained manipulation, we did not select them for comparison because some of their functions require bimanual gestures. For instance, to scale up an object using tBox, users have to drag two opposite edges on the same face. When the display size of tBox is large, it is uncomfortable to reach two opposite edges using fingers of one hand.

5.1 Hypotheses

We have made two hypotheses before conducting the controlled user study:

H1: The Constraint Menu technique is more efficient than the two other technique, because the coordination of two hands simplifies specifying the manipulation constraint, and reduces the burden of the DH.

H2: The Constraint Menu technique requires less camera navigation than the other two techniques because its performance is less affected by the viewpoint of the camera.

5.2 Apparatus

The experiment was conducted on an Apple iPad Air 2, with a 9.7" touchscreen. The screen has a resolution of 2048×1536 pixels. Content was displayed in Portrait mode. The experimental environment was developed using Unity 3D with C#.

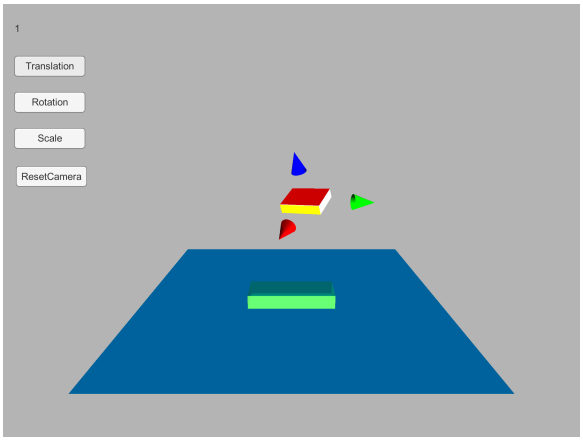


Figure 8: *The environment of the evaluation.*

5.3 Participants

Twelve unpaid subjects (10 males and 2 females, 20 to 27 years old) participated in this study. They were all students from a university. All of them use smartphones frequently in everyday life but only 4 of them use frequently 3D modeling software. Only one participant is left handed. For the left handed participant, the Constraint Menu and buttons of the Widget-based technique was positioned on the right-hand side of the screen. For other participants, these interactive UI widgets were displayed on the left-hand side.

5.4 Procedure and Design

Similar to [Martinet et al. 2010],the participants were asked to perform a docking task using the three manipulation techniques. As shown in Figure 8, at the beginning of each trial, a cuboid was displayed in a random position in the scene and another semi-transparent cuboid that serves as a target was fixed on the surface. For each trial, participants had to move the cuboid towards the target. A trial was considered to be complete when the cuboid was moved close enough to the target position, orientation and scale. Indeed, once the difference in position, orientation and scale were under a pre-defined threshold value, the trial was finished and a start button was displayed for launching the next trial. To avoid orientation ambiguity, one face of the cuboid was colored in red and another face was colored in yellow. The red face should coincide with the top of the target and the yellow face should coincide with the front side of the target. The initial position of the cuboid was generated randomly on a semi-sphere whose center was the target. The radius of the sphere was set to a fixed value to ensure that the cuboid was inside the initial viewpoint of the camera at the beginning of the task. The initial difference in orientation between the cuboid and the target was 120° around a random vector. The initial scale difference was generated using the equations (1) (2) and (3). (Sx_c, Sy_c, Sz_c) is the scale of the cuboid in the local coordinate system while (Sx_t, Sy_t, Sz_t) is the scale of the target. (Sx_o, Sy_o, Sz_o) is the scale difference between the cuboid and the target and (Sx_v, Sy_v, Sz_v) is the random scale direction. The scale range is set to 0.5. The target was generated at a fixed position for each trial. During the experiment, the camera could be manipulated freely to change the viewpoint; however, the starting point of view was the same at the beginning of each trial. Moreover, we added a button to reset the camera viewpoint to avoid losing the cuboid and the target when the camera is translated or rotated too much. When this button was pressed during one trial, this trial was considered to

be invalid and the participant had to start it over.

$$(Sx_c, Sy_c, Sz_c) = (Sx_t * Sx_o, Sy_t * Sy_o, Sz_t * Sz_o) \quad (1)$$

$$(Sx_o, Sy_o, Sz_o) = (1+0.5*X_v, 1+0.5*Y_v, 1+0.5*Z_v) \quad (2)$$

$$\sqrt{X_v^2 + Y_v^2 + Z_v^2} = 1 \quad (3)$$

In this within-subjects design experiment, the only independent variable was the manipulation technique (TECH) with three levels; Widget-Based, Duo-Finger Constraint and Constraint Menu. To eliminate any effect of the trial order, TECH order was counterbalanced. Because 3 techniques were compared, there were 6 possible experimental orders. Twelve participants were divided into 6 groups and the permutation order was the same in each group. For each TECH, participants were asked to perform 10 manipulation trials. A total of $12 \times 3 \times 10 = 360$ manipulation trials were performed in this experiment. Before the experimental sessions for each technique, instructions were given to participants on how to manipulate objects and they had 5 to 15 minutes to get familiar with the technique. To reduce the learning effect, participants were asked to perform at least 5 and at most 10 docking trials for each technique before the formal experiment. After the experiment, participants were asked to fill in a questionnaire to evaluate each technique.

6 Results

6.1 Completion time

The completion time is defined as the time it takes to transform the cuboid to the target. For each task, the record of the completion time started from the moment when the start button was clicked and ended when the start button was displayed again for the next task.

A one-way repeated-measures ANOVA was used to compare the performance of the different techniques. Results show a significant main effect for TECH ($F_{(2,10)} = 6.573, p = 0.015$). The mean completion time was $90.83s$ for Widget-based, $86.08s$ for Duo-Finger Constraint and $75.31s$ for Constraint Menu (Figure 9). The post-hoc tests with Bonferroni correction show that Widget-based was significantly slower than Constraint Menu ($p = 0.01$). No other significant differences were observed. Therefore, the first hypothesis is only partially validated.

Besides the total completion time, we have also compared the translation, rotation, scale and camera navigation times among the three techniques (Figure 9). The one-way repeated-measures ANOVA show a significant main effect of TECH for the rotation time ($F_{(2,10)} = 11.525, p = 0.003$), scale time ($F_{(2,10)} = 9.507, p = 0.005$) and camera navigation time ($F_{(2,10)} = 6.360, p = 0.017$). The post-hoc tests with Bonferroni correction show that Widget-based was significantly slower than Constraint Menu for rotation ($p = 0.001$), scale ($p = 0.003$) and camera navigation ($p = 0.02$). Widget-Based was also significantly slower than Duo-Finger Constraint for scaling ($p = 0.022$). Duo-Finger Constraint was significantly slower than Constraint Menu for rotation ($p = 0.02$) and camera navigation ($p = 0.042$). This validates our second hypothesis. No other significant differences were observed.

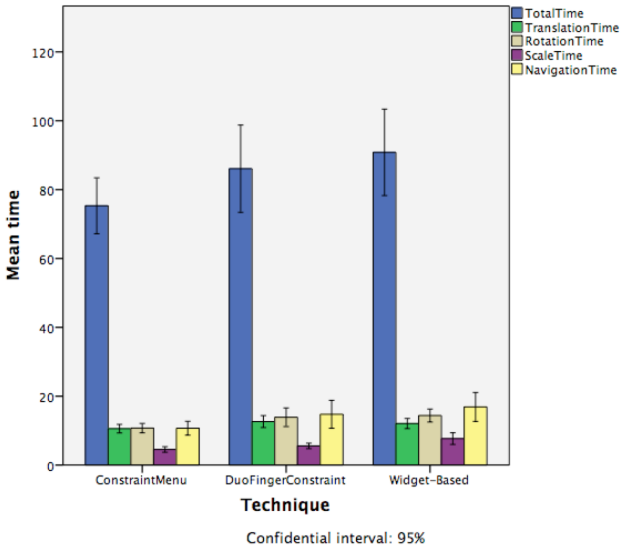


Figure 9: *Mean completion time of different techniques. The translation, rotation, scaling and navigation mean time is also displayed.*

6.2 Translation & Rotation Coordination

In addition, we have calculated the translation and rotation coordination coefficient to compare the translation and rotation efficiency. Similar to [Martinet et al. 2010], the translation coordination coefficient is defined as the ratio of the length of shortest path and the length of actual path. The rotation coordination coefficient is defined as the ratio of the initial rotation mismatch and the amount of actual rotation.

The one-way repeated-measures ANOVA show a significant main effect of TECH for both translation coordination ($F_{(2,10)} = 17.943, p < 0.0001$) and rotation coordination ($F_{(2,10)} = 16.762, p = 0.001$) among the different techniques. The mean translation coordination was 0.51 for Widget-Based, 0.43 for Duo-Finger Constraint and 0.51 for Constraint Menu. The mean rotation coordination was 0.35 for Widget-Based, 0.445 for Duo-Finger Constraint and 0.493 for Constraint Menu (Figure 10). The post-hoc tests with Bonferroni correction show that Duo-Finger Constraint had a significantly lower translation coordination value than both Widget-based ($p = 0.002$) and Constraint Menu ($p < 0.0001$).

6.3 Subjective evaluation

After accomplishing manipulation tasks for each technique, participants were asked to answer a questionnaire to evaluate the technique regarding six different criteria using a seven point Likert scale (1-very bad, 7-very good). We also asked them to sort all the techniques according to their preference. Seven participants have chosen Constraint Menu as the preferred technique and four participants have selected Duo-Finger Constraint. Table 1 shows the answers for the different criteria. The criterion of intuitiveness is about whether the metaphor of the technique is appropriate and is consistent with the manipulation. The criterion of fluidity is about whether manipulations can be performed without being interrupted by other operations. The Friedman test shows that there was no significant difference among the different techniques for preference of translation, rotation and intuitiveness.

On the other hand, a significant difference exists among different

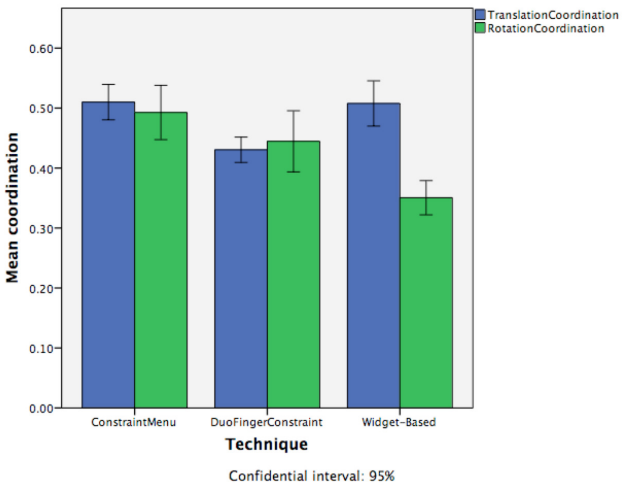


Figure 10: *Mean translation and rotation coordination of different techniques.*

	Widget-Based	Duo-Finger Constraint	Constraint Menu
Translation	5.6 (0.9)	5.5 (1.0)	5.8 (0.7)
Rotation	5.6 (1.3)	4.7 (1.3)	5.5 (1.1)
Scale *	4.1 (1.5)	5.5 (1.1)	5.7 (1.2)
Efficiency *	5.1 (1.2)	4.8 (1.0)	5.7 (1.2)
Intuitiveness	5.3 (1.2)	4.5 (1.1)	5.1 (0.9)
Fluidity *	4.4 (1.5)	4.8 (0.9)	5.4 (0.9)

Table 1: *The result of subjective evaluation. Significant differences were found for scaling, efficiency and fluidity.*

techniques for scaling ($Chi - Square = 12.905, p = 0.002$). The Wilcoxon Signed Rank test shows that participants agreed that Widget-Based was more difficult to use for scaling than Duo-Finger Constraint ($Z = -2.803, p = 0.005$) and Constraint Menu ($Z = -2.699, p = 0.007$). Regarding manipulation efficiency, a significant difference was found among the different techniques ($Chi - Square = 8.579, p = 0.014$). The Wilcoxon Signed Rank test show that participants strongly preferred Constraint Menu to Duo-Finger Constraint ($Z = -2.810, p = 0.005$). A significant difference was also found among the different techniques for fluidity ($Chi - Square = 6.061, p = 0.048$). The Wilcoxon Signed Rank test show that participants strongly agreed that Constraint Menu was more fluent than Widget-based ($Z = -2.308, p = 0.021$).

6.4 Discussion

We have conducted a controlled experiment to examine whether our technique helps to simplify object manipulation on touch-based mobile devices. On the one hand, we wanted to investigate whether our technique outperforms the other two manipulation techniques regarding the completion time and manipulation coordination. On the other hand, we wanted to know how users would subjectively evaluate our technique.

The results indicate that there was a performance gap between Widget-Based and Constraint Menu. The significant difference mainly came from the performance difference in rotation, scale and navigation tasks.

The difference in rotation performance can be partly explained by the fact that different finger movements were used. In fact, the Widget-Based technique requires rotating the finger around the controlled object while the Constraint Menu technique maps linear finger movements to object rotation. Our hypothesis was that making precise control is easier when using linear movements than circular movements. Moreover, when rotating the widget, the object was occasionally occluded by the stretched finger. This affected the rotation precision and participants had to take more time to adjust the object orientation. Because our technique permits making touch inputs in the empty space, the occlusion problem can be avoided. In addition, when the principal axis is parallel to the screen, the rotation algorithm could not infer the rotation intention correctly due to lack of depth information. In this case, moving the widget may produce undesired rotations. This can also explain why the Widget-Based technique had significantly lower value of rotation coordination than the other two techniques.

For object scaling, the Widget-Based technique was more affected by the point of view. In fact, for some viewpoints, the scaling widgets were hidden. We observed that some users were changing repetitively the viewpoint in order to modify the scale and then coming back to the original viewpoint to examine the result. This increased the time for both scaling and navigation. Sometimes, when the camera was far from the object and the display size of the object became smaller, undesired scaling widgets were touched. Users had to make touch inputs more carefully to avoid triggering undesired scaling operations. For Duo-Finger Constraint and Constraint Menu only the specified axis or plane was displayed. Hence, users were aware of whether the desired constraint was selected before performing the pinch gesture to modify the scale. Because both rotation and scaling of Widget-based were influenced by the perspective, users had to perform more camera movements to adjust the point of view. This can explain why the Widget-Based technique consumed more navigation time than the Constraint Menu technique.

The results also indicate that Duo-Finger Constraint had significantly slower rotation and navigation time than Constraint Menu. We observed that some participants preferred checking the orientation mismatch when the camera faced directly the side faces of the object. However, one limitation of Duo-Finger Constraint is that the object could not be rotated around the axis that is perpendicular to the screen. As a result, they changed the camera viewpoint to adjust the orientation and then came back to check the result. For Constraint Menu, users could rotate the rotation widget around the object when facing directly the side face. This helped them saving both rotation and navigation time.

One interesting finding is that although no significant difference of translation time exists among the different techniques, Duo-Finger Constraint had significantly less translation coordination than the two other techniques. Because in some camera perspectives the projection of two axes overlapped each other, it was difficult to select the right one using two fingers. As a result, sometimes, undesired translations were triggered. We think that the longer translation path of Duo-Finger Constraint is duo to translations along the undesired axis.

For subjective evaluation, we have found significant differences for scaling, efficiency and fluidity. Some participants said that it is more likely to trigger undesired scaling actions in some camera perspectives when using the Widget-based technique. They thought that the scaling methods of the other two techniques are easier to use. This comment is consistent with our quantitative results and observations.

However, we did not expect Duo-Finger Constraint to get the lowest preference score regarding efficiency since Duo-Finger Constraint required less mean completion time than Widget-Based. Some users pointed out that the performance of Duo-Finger Constraint

required switching the wrist angle frequently to select the desired axis and it was uncomfortable to perform touch inputs when the wrist was twisted. Instead of adjusting the wrist angle, some participants rotated frequently the tablet with the NDH to simplify axis-constraint selection. In addition, some participants also commented that they had to make more camera navigation to avoid the overlap of two axes.

Participants strongly agreed that Constraint Menu was more fluent than Widget-based. This can be explained by the fact that in Widget-based, participants had to press function buttons frequently to switch operation modes. Once a constraint was selected, Constraint Menu allowed users to switch operation modes more seamlessly by changing the gesture of the DH.

Some participants have also pointed out some limitations of our technique. First, they said that the rotation gesture is less intuitive to perform than the translation and scaling gestures. In fact, it is less intuitive to determine in which direction the finger should be dragged to rotate an object. They found that the rotation arrow was misleading when its projection on the screen was parallel to that of the axis. Participants tended to follow the direction of the arrow to trigger object rotation. However, since the finger was dragged in the direction parallel to the axis, object translation was triggered. To overcome this issue, we propose two potential solutions. The first one is to refresh the position of the arrow when the constraint is specified to ensure that the projection of the arrow is perpendicular to the axis. Thus, the visual hint can help users drag the finger in the correct direction. The second solution is to use the two joint fingers to trigger the rotation task. We will examine whether these two potential methods can effectively reduce the ambiguity in a future work.

Another problem we have observed is that the use of the Constraint Menu is limited by the hand size and the holding posture. One participant with large hands said he had to tilt the fingertip of the thumb to avoid pressing unwanted constraint options. Two participants preferred displaying the Constraint Menu higher since they were used to hold the tablet at a higher position. It seems necessary to provide the possibility to adjust the position and size of the menu to adapt to personal preference.

7 Conclusion & Future Work

In this paper, we described a new manipulation technique for touch-based mobile devices. Our technique uses the NDH to set the manipulation constraint and the DH to trigger the operation and control transformation. With the help of the Constraint Menu, axis- and plane-constraints can be specified easily and touch inputs of the dominant hand are not restricted by the position, orientation and size of objects. In addition to the manipulation technique, we have also proposed an interface which allows users to make rapid 3D sketching. Our interface provides a set of common-used functions such as face manipulation and boolean operations. We have redefined their interaction technique to adapt it to the touch-based paradigm. To evaluate the performance of our manipulation technique, we have conducted a controlled experiment in which we have compared our technique with two state-of-the-art manipulation techniques. The results indicate that our technique outperforms the Widget-Based technique regarding the efficiency.

One limitation of our technique is that RST manipulation could not be made along an arbitrary axis. Because touchscreens only accept 2D inputs, it is difficult to specify an arbitrary axis in 3D space using touch inputs. We are interested in this topic and will try to look for a solution for this problem in the future. In this study we have not compared our technique with tBox, Toucheo and LTouchIt because some bimanual gestures of these techniques are not practical to perform on TabletPCs. However, we plan to implement the Constraint Menu technique on a large fixed surface and conduct a more comprehensive evaluation to compare it with these techniques. Although we have developed a new touch-based interface for 3D sketching, we have not evaluated it in this work. In the future, we are planning to make a user study to find out whether our interface can help expert users to reduce the sketching time and improve the work fluidity on mobile devices.

References

AU, O. K.-C., TAI, C.-L., AND FU, H. 2012. Multitouch gestures for constrained transformation of 3d objects. In *Computer Graphics Forum*, vol. 31, Wiley Online Library, 651–660.

BONNET, D., APPERT, C., AND BEAUDOUIN-LAFON, M. 2013. Extending the vocabulary of touch events with thumbrock. In *Proceedings of Graphics Interface 2013*, Canadian Information Processing Society, 221–228.

BORING, S., LEDO, D., CHEN, X., MARQUARDT, N., TANG, A., AND GREENBERG, S. 2012. The fat thumb: using the thumb's contact size for single-handed mobile interaction. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*, ACM, 39–48.

CAO, X., WILSON, A. D., BALAKRISHNAN, R., HINCKLEY, K., AND HUDSON, S. E. 2008. Shapetouch: Leveraging contact shape on interactive surfaces. In *Horizontal Interactive Human Computer Systems, 2008. TABLETOP 2008. 3rd IEEE International Workshop on*, IEEE, 129–136.

COHÉ, A., DÈCLE, F., AND HACHET, M. 2011. tbox: a 3d transformation widget designed for touch-screens. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 3005–3008.

DE ARAÚJO, B. R., CASIEZ, G., JORGE, J. A., AND HACHET, M. 2013. Mockup builder: 3d modeling on and above the surface. *Computers & Graphics 37*, 3, 165–178.

GUIARD, Y. 1987. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of motor behavior 19*, 4, 486–517.

HACHET, M., BOSSAVIT, B., COHÉ, A., AND DE LA RIVIÈRE, J.-B. 2011. Toucheo: multitouch and stereo combined in a seamless workspace. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, ACM, 587–592.

HANCOCK, M., CARPENDALE, S., AND COCKBURN, A. 2007. Shallow-depth 3d interaction: design and evaluation of one-, two-and three-touch techniques. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 1147–1156.

KIN, K., MILLER, T., BOLLENSDORFF, B., DEROSE, T., HARTMANN, B., AND AGRAWALA, M. 2011. Eden: a professional multitouch tool for constructing virtual organic environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 1343–1352.

LIU, J., AU, O. K.-C., FU, H., AND TAI, C.-L. 2012. Two-finger gestures for 6dof manipulation of 3d objects. In *Computer Graphics Forum*, vol. 31, Wiley Online Library, 2047–2055.

MARTINET, A., CASIEZ, G., AND GRISONI, L. 2010. The effect of dof separation in 3d manipulation tasks with multi-touch dis-

plays. In *Proceedings of the 17th ACM Symposium on Virtual Reality Software and Technology*, ACM, 111–118.

MENDES, D. 2011. *LTouchIt: LEGO Modeling on Multi-Touch Surfaces*. PhD thesis, Masters thesis, Instituto Superior Técnico.

PACZKOWSKI, P., DORSEY, J., RUSHMEIER, H., AND KIM, M. H. 2014. Paper3d: bringing casual 3d modeling to a multi-touch interface. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, ACM, 23–32.

REISMAN, J. L., DAVIDSON, P. L., AND HAN, J. Y. 2009. A screen-space formulation for 2d and 3d direct manipulation. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, ACM, 69–78.

ROUDAUT, A., LECOLINET, E., AND GUIARD, Y. 2009. Microrolls: expanding touch-screen input vocabulary by distinguishing rolls vs. slides of the thumb. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 927–936.

SCHMIDT, R., SINGH, K., AND BALAKRISHNAN, R. 2008. Sketching and composing widgets for 3d manipulation. In *Computer Graphics Forum*, vol. 27, Wiley Online Library, 301–310.

SUN, Q., LIN, J., FU, C.-W., KAIJIMA, S., AND HE, Y. 2013. A multi-touch interface for fast architectural sketching and massing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 247–256.

WAGNER, J., HUOT, S., AND MACKAY, W. 2012. Bitouch and bipad: designing bimanual interaction for hand-held tablets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 2317–2326.

WILSON, A. D., IZADI, S., HILLIGES, O., GARCIA-MENDOZA, A., AND KIRK, D. 2008. Bringing physics to the surface. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, ACM, 67–76.

WILSON, A. D. 2009. Simulating grasping behavior on an imaging interactive surface. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ACM, 125–132.