Pers Ubiquit Comput (2013) 17:1551–1572
DOI 10.1007/s00779-012-0594-2

ORIGINAL ARTICLE

Dual-Finger 3D Interaction Techniques for mobile devices

Can Telkenaroglu · Tolga Capin

Received: 9 March 2012 / Accepted: 31 July 2012 / Published online: 4 September 2012

Abstract Three-dimensional capabilities on mobile devices are increasing, and the interactivity is becoming a key feature of these tools. It is expected that users will actively engage with the 3D content, instead of being passive consumers. Because touch-screens provide a direct means of interaction with 3D content by directly touching and manipulating 3D graphical elements, touch-based interaction is a natural and appealing style of input for 3D applications. However, developing 3D interaction techniques for handheld devices using touch-screens is not a straightforward task. One issue is that when interacting with 3D objects, users occlude the object with their fingers. Furthermore, because the user's finger covers a large area of the screen, the smallest size of the object users can touch is limited. In this paper, we first inspect existing 3D interaction techniques based on their performance with handheld devices. Then, we present a set of precise *Dual-Finger 3D Interaction Techniques* for a small display. Finally, we present the results of an experimental study, where we evaluate the usability, performance, and error rate of the proposed and existing 3D interaction techniques.

Keywords Mobile 3D environments · Touch-screens · Multi-touch input · Dual-finger techniques

C. Telkenaroglu (✉) · T. Capin
Department of Computer Science,
Bilkent University, Bilkent, 06800 Ankara, Turkey
e-mail: cant@cs.bilkent.edu.tr

T. Capin
e-mail: tcapin@cs.bilkent.edu.tr

1 Introduction

Today, the popularity of 3D media in mobile devices is increasing, and handheld devices with 3D capabilities are becoming common. Graphics hardware support for OpenGL ES in mobile devices opens up new possibilities for the 3D user experience as well as applications such as 3D gaming, 3D maps, and data visualization. Three-dimensional user interfaces (UI) and applications such as shared virtual environments offer the possibility of utilizing the small display area of a mobile device in an efficient manner. The limitations of the mobile context, including the small physical screen size and limited input modalities, can, to a degree, be overcome with 3D interaction. The emerging output solutions, such as autostereoscopic displays that do not require special glasses to achieve a stereoscopic effect, also have the potential to significantly change the user experience for future 3D mobile applications.

Interactivity is a key feature of the 3D user experience with mobile devices. It is hoped that users will actively engage with the 3D content, instead of being passive consumers. A number of user input alternatives currently exist on mobile devices, including the use of touch-screen-based inputs, inertial trackers, and camera-based tracking; each with advantages and disadvantages. Among them, multi-touch interfaces have emerged as the standard input technique. Because touch-based interaction provides a direct means of interacting with 3D content, it is also a natural and appealing style of input for 3D applications. Inertial trackers, such as three-axis acceleration sensors and gyroscopes for rotational sensing, also have the potential to increase the richness of interaction with handheld devices.

Three-dimensional interaction techniques have been extensively studied in immersive virtual environments,

with the use of head-mounted displays and tracking devices such as data gloves, and on desktop VR configurations with a keyboard and mouse. Several researchers have studied 3D interaction techniques that approach the richness of reality, particularly for desktop and large-scale interactions. Shneiderman [1] examines the features for increasing the usability of 3D user interfaces primarily for desktop and near-to-eye displays and proposes general guidelines for UI developers. These guidelines include: better use of depth cues, particularly occlusion, shadows, and perspective; minimizing the number of navigation steps in the UI; improving text readability with better rendering; taking into account the limited angle of the view position, contrasting with the background, among others. Bowman et al. analyze interaction techniques common in 3D user interfaces and develop a taxonomy of universal tasks for interacting with 3D virtual environments: selection and manipulation of virtual objects; travel and way finding within a 3D environment; issuing commands via 3D menus; and symbolic input such as text, labels, and legends. Defining appropriate 3D interaction techniques is still an active field [2].

Although touch-based interaction provides a direct means of interacting with 3D content, developing 3D interaction techniques for handheld devices with multi-touch displays is not a straightforward task. Due to the small size of the device, the area of interaction and display is limited. Interacting with a 3D object using multi-touch input, users often occlude the object with their fingers [3]. With the increasing complexity of 3D scenes, this limitation becomes a major issue. Another problem is the area that the user's finger covers on the screen; the smallest size of the objects that users can touch on the screen is also limited. Therefore, it is difficult to perform a precise, pixel-level selection in dense or cluttered environments with varying object sizes [4].

In this paper, we first inspect existing 3D user interaction techniques and present a qualitative evaluation based on their performance when applied to handheld devices. Second, we present a new set of precise 3D interaction techniques, which includes *Dual-Finger Navigation* for navigation tasks, *Dual-Finger Midpoint Ray-Casting* and *Dual-Finger Offset Ray-Casting* for 3D object selection tasks, and *Dual-Finger Translation* and *Dual-Finger Rotation* for 3D object manipulation tasks. These techniques are inspired by the *Dual-Finger Midpoint* and *Dual-Finger Offset* techniques [4], and we extend this approach to interaction tasks in a 3D environment. Finally, we present the results of a controlled user experiment where we evaluate the performance of the existing and proposed 3D interaction techniques on handheld devices.

2 Related work

The primary principle of 3D virtual environments is to provide the user a feeling of presence. This can be obtained through natural and realistic interaction techniques with the environment.

2.1 3D user interaction techniques

Several 3D interaction techniques have been proposed for virtual environments in the past two decades, and these are generally classified under the "universal tasks" of navigation, manipulation/selection, system control, and symbolic input. Research in this field addresses such issues as the empirical design and evaluation of displays, design and evaluation of novel interaction techniques, and design of input devices and their mapping to 3D interaction [2].

Selection and Manipulation Techniques can be classified with respect to the task that is carried out, and the metaphors used in them [2, 5]. Selection techniques are composed of a sequence of two subtasks: indicating the target object and the optional subtask of confirming the selection. As a result, the user receives feedback indicating that the object is selected. Indication of the target object can be performed by occluding the object, touching the object in the image space, or pointing. Considering the taxonomy based on the metaphors, selection and manipulation techniques have been classified as egocentric and exocentric [5]. Exocentric techniques, such as *World-in-Miniature* or *Automatic Scaling of the World*, use an external view of the environment and represent the position and orientation of the user in the scene [2, 6]. Egocentric techniques include *Virtual Hand Metaphor*-based techniques such as *Virtual Hand* and *Go–Go*; as well as *Virtual Pointer Metaphor*-based techniques such as *Ray-Casting*, *Aperture*, *Flashlight*, and *Image-Plane* [2, 7–9]. To perform a selection in the virtual world, pointing techniques are generally considered more precise than virtual hand-based techniques, because precisely controlling a virtual hand cursor in 3D space is more difficult. Virtual hand techniques generally perform more effectively for object manipulation tasks because they are able to provide appropriate feedback to the user. Hybrid interaction techniques are also possible such as Bowman et al.'s *HOMER* technique [8].

Navigation techniques can be classified in different ways. One approach is to classify the navigation as *active* (controlled by the user), *passive* (controlled by the system), or *semi-automated* (the system controls the movement, but the user explores the travel path) [2]. Another classification approach considers the physical state of the user. For example, if the user moves physically in the real world to navigate in the environment, this is called a *Physical Technique*. On the other hand, if the user remains

stationary but controls the movement and rotation via an input device, that technique is classified as a *Virtual Technique*. A hybrid method allows one subtask to be performed physically and the other virtually. A third classification of navigation techniques uses a task-based taxonomy, with secondary consideration for the level of user control [10].

The navigation task can be decomposed into subtasks of *rotation* and *movement*. A recent study by Han et al. [11] offers variants of the *Possession* metaphor and *Rubberneck Navigation*. In the first technique, the user can select an object to have the object's field of view. The second technique overcomes the problem of using separate mechanisms for movement and camera rotation. The user moves the mouse to look around, then holds the mouse button, and draws a path to move along that path. The same study proposes another technique called *Speed-Coupled Flying with Orbiting*. Users move the mouse left and right for camera rotation, and front and back for travel. When the user drags the mouse more quickly, the camera gains altitude.

When larger display sizes than on a handheld device are used, it is possible to use the whole hand or both hands to control navigation. In a recent study, Wu et al. [12] present a multi-touch technique, where two fingers bring out the *Powers of Ten Ladders* and another finger from the second hand slides along the ladder to exponentially increase the camera distance from the center of the 3D environment. This technique also rotates the camera around the y axis by left/right slides of the hand, around the x axis using up/down slides, and around the z axis with clockwise and counterclockwise motion.

A number of studies focus on a special case of navigation: *panning the camera* around a selected object. One study presents a 3D widget called *Navidget*, which uses a ray that is cast to indicate a focus area, to be covered with a half sphere carried at the end of the ray [13]. If the ray intersects an object, the sphere snaps to it to make this task more controlled. In the next step, the user places the camera at the spherical coordinate hit by the ray. Another recent mobile interaction study shows that having a controlled camera-panning approach will prevent users from getting lost [14]. This technique maps between touch-screen finger movements, to achieve a certain amount of controlled camera rotation to prevent disorientation.

Isomorphism is also an issue of usability for these techniques. Isomorphic interaction techniques use one-to-one mapping between the physical world, where input is performed, and the virtual world. Such techniques generally feel more natural to the user but are not as comfortable to use. Non-isomorphic techniques take advantage of performing a mapping between the user's inputs in the physical world and action in virtual world [2]. According to the guidelines offered by Bowman et al., tasks with a low cognitive load and that need less physical effort from the user, such as short rotations, should be performed physically [2]. It is possible to implement navigation techniques on mobile devices *physically* through acceleration sensors, such as directing the view point, and *virtually* using touch-screen gestures to rotate the view and move the camera. Hürst et al. compare virtual and physical rotation and report that physical rotation is more appealing to 80 % of the test subjects and a better choice through which to perceive the environment [15].

2.2 3D interaction with multi-touch displays

Multi-touch 3D interaction with 2D displays has recently gained interest, particularly on tabletop displays. Tabletop 3D interaction studies focus mainly on object manipulation tasks, as navigation tasks do not map naturally to the tabletop environment, and selection tasks are mapped straightforwardly on the exocentric and large-display view of these applications. Because this new generation of hardware more closely emulates physical workspaces, various approaches are proposed for physical interaction with 3D content.

Wilson et al. propose the use of proxy objects to model rich physical tabletop interactions with 3D objects, such as pushing, grabbing, pinching, and dragging [16]. Hilliges et al. [17] build a tabletop system, based on a depth camera and holoscreen that senses movement up to 0.5 m above the tabletop, which enables richer interactions above the table screen. These techniques are limited to the tabletop metaphor, however, and not suitable for 3D virtual environment interaction on general-purpose multi-touch displays, such as mobile devices. The BumpTop environment, which uses a physics engine to add realism to the tablet PC desktop, supports features such as collisions, mass, and piling [18]. However, this method is based on a single point of view, which never changes, and uses menu-based interaction, which limits the 3D capability.

Recently, a number of studies focus on the particular problem of mapping the user's 2D input to 3D objects on a tabletop display. Hancock et al. [19] demonstrate one-, two-, and three-fingered rotation-and-translation control techniques by mapping 2D input to 3D object manipulation. One conclusion from the user studies in this work is that rotation and translation tasks can be separated, which provides a natural interface for communication without sacrificing performance. This method requires learning special gestures, defined by a specific order of touching with different fingers, that the authors state is natural for users to learn. Another recent work for direct multi-touch interaction is Reisman's method [20]. This approach solves constraints set by the user's fingers, which minimizes the

error between the screen-space projection of contact points and their target positions. Martinet et al. [21] evaluate these two methods for their integrality and separability properties by a controlled user experiment: whether separation of translation and rotation in these techniques affect 3D performance for object manipulation. They conclude that separation of different degrees of freedom (DOF) affect manipulation performance, and this work proposes a new screen-space solution.

Special-purpose UI widgets have recently been proposed for object manipulation. Fabrice et al. [22] present a widget called *tBox* to offer a physical gesture metaphor for manipulating a selected object on a multi-touch screen. This widget is viewed as the object's bounding box and supports rotation by controlling its inertia, translation via sliders on the edges of the widget, and scaling the object through pinch gestures with fingers. Henrysson et al. [23] compare using keypad buttons and one-handed physical movement of a phone to move the selected object in an augmented-reality environment. A user experiment reveals that positioning the object is more natural and faster using physical movement than using the buttons. The same study compared the *arc-ball* technique, keypad input, and physical interaction to rotate the selected object. The user study showed that physical interaction was easiest to use and the most accurate, and *arc-ball* was fastest, although hardest to control. Lastly, Martinet et al. [24] propose a method called *Z-Technique*, which uses one finger to move the selected object in the image-plane and two fingers moved in the same direction to control the object in depth.

2.3 Precise touch-screen interaction

With touch-screen-based interaction in mobile devices, efficient use of screen space is essential. For touch-screen-based UIs, the main limitation is that interactive elements must be presented in at least 1 × 1 cm square on the touch surface in order to be comfortably picked by an average finger [25]. This fact limits how many UI elements can be rendered in the display. A possible solution to this problem is to layer the elements in the 3D scene, such that the elements are large enough to support finger-touch input in the top layer, but denser in the underlying layers. This solution, however, increases clutter in the scene and limits 3D user interaction capabilities in 3D applications.

Various techniques are proposed for precise selection in 2D interfaces. Benko et al. [4] posit precise 2D selection techniques that overcome the problem of finger occlusion on the screen: *Dual-Finger Offset* and *Dual-Finger Midpoint*. The first technique offsets the cursor to the midpoint when a second finger is placed on the screen. After the second finger is removed, the cursor moves with that offset prior to the primary finger. In the second technique, the secondary finger is never removed from the screen, and the cursor is at the midpoint of the fingers. Since the 2D cursor is at the midpoint of the fingers, which cover an area of 1 cm^2 on the screen, geometrically it is not possible to select an object on the corners of the screen without scrolling. Therefore, this method limits 2D target selection from the screen corners.

3 Qualitative evaluation of 3D interaction techniques

The 3D interaction techniques mentioned in Sect. 2.1 are primarily designed for immersive or desktop PC environments. This section compares some of the well-known 3D interaction techniques in terms of their applicability to handheld devices with multi-touch displays, inspired by

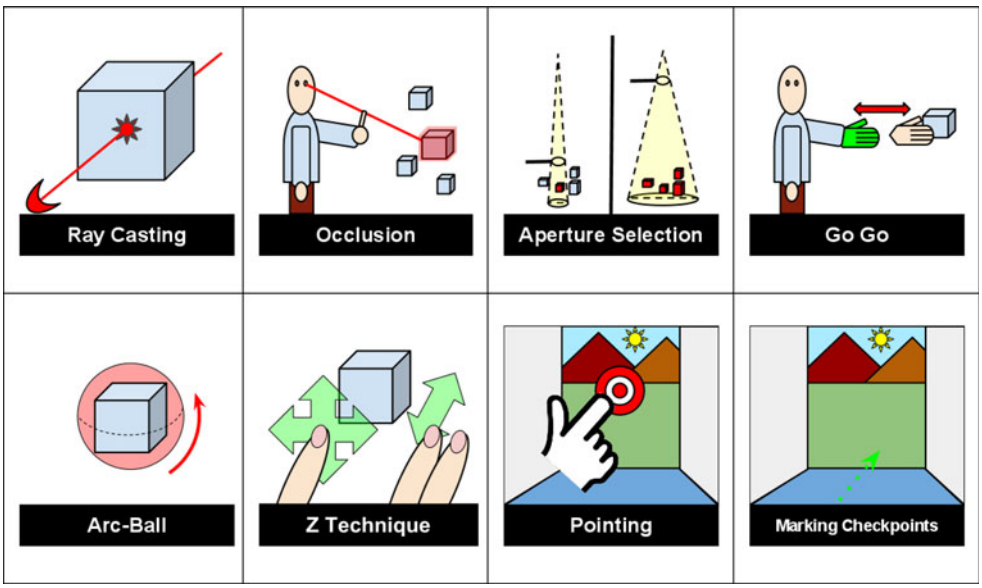

Fig. 1 3D interaction techniques investigated while building the evaluation. *First row*, *left* to *right*: *Selection* techniques Ray-Casting, Occlusion, Aperture Selection, Go–Go. *Second row*, *left* to *right*: *Manipulation* techniques Arc-Ball widget used for rotation, Z-Technique for positioning. *Navigation* techniques: pointing, marking checkpoints [2]

Bowman et al.'s [26] formalization principles. As an indicator of performance, for each 3D user interaction task (Fig. 1), we outline a number of factors that influence the interaction's effectiveness on mobile devices.

3.1 Selection techniques

It is difficult to compare different popular selection techniques for the handheld environment because most techniques are designed for input devices and usage environments other than the mobile context, and their performance in multi-touch displays has not been evaluated. Therefore, we first outline a number of factors that affect the performance of these interaction techniques in the mobile context of use.

3.1.1 Object size/distance

These two attributes are related to the geometric area covered by the object on the screen. When the object is small or has a higher depth value, the selection technique must be sufficiently precise. Techniques based on ray shooting, such as *Ray-Casting* or *Occlusion Techniques*, have high performance in selecting objects in immersive environments, unless the objects are small-sized or distant. With *Ray-Casting*, the user shoots a ray to the virtual scene using a pointer to the screen, whereas with *Occlusion Technique*, the user selects the target object using a finger or marker in a way that will occlude the object from the perspective of the user [3]. The *Aperture* technique uses a volumetric cone with the top of it at the user's view point and that goes through a circular marker held by the user at a further level. This technique effectively selects small objects [2, 9] and has higher precision when the marker is further from the eye, which results in a cone with a smaller base radius. The *Go–Go* selection technique, based on the virtual hand metaphor, has a different approach from ray-based techniques [8]. With this technique, the user physically selects objects using an electronic glove as an input device. The length of the virtual arm can be adjusted to scale the distance to select further objects with ease.

3.1.2 Density

Virtual environments may contain a large number of tightly grouped objects, which results in a dense environment. In such environments, selection requires a more precise technique. *Ray-Casting* is reported to perform effectively in dense environments in immersive or desktop contexts [2]. The *Aperture* technique, although effective at selecting small or distant objects, performs less precisely in a dense group of objects [7]. The *Occlusion* technique requires an object specifier, for example a finger or stylus in the mobile context of use. Due to high occlusion with this tool compared to virtual objects, performance decreases with a dense group of small or distant objects, which is an important issue for mobile displays [2]. The *Go–Go* technique is also expected to have low performance when selecting objects in dense environments [2].

3.1.3 Occlusion

In any environment, objects usually, partially, or fully occlude each other. Under these conditions, *Ray-Casting*, *Aperture*, and image-plane technique *Occlusion* cannot select fully occluded objects. On the other hand, since *Ray-Casting* has greater precision, it selects objects that are partially occluded in desktop environments [2]. The *Go–Go* technique can easily select highly occluded objects, and even those objects completely occluded by other transparent objects [2, 9].

Table 1 presents an evaluation of the standard 3D selection in terms of the above factors. The rating ranges from "−" for low selection performance to "++" for the most effective performance.

3.2 Manipulation techniques

Following the recent findings in the field [19, 21], we propose a separate discussion for ease of positioning and ease of rotation in 3D manipulation tasks. Table 2 summarizes the compared manipulation techniques in this study.

3.2.1 Ease of translation

The first subtask of manipulation is to reposition the object in the virtual environment. Two physical techniques, *Ray-Casting* and *Go–Go*, provide the most effective performance for translation based on the physical translation of the input devices. However, *Ray-Casting* cannot move the object along the z axis, and *Go–Go* is more effective in positioning objects [2, 8, 9]. The *Z-Technique*, a virtual technique targeted to multi-touch displays, is expected to provide an effective manipulation method for translation [24]. In this technique, the user moves the object on the vertical plane using his one finger and adjusts the depth of the object by moving his two fingers up and down on the touch-screen.

3.2.2 Ease of rotation

The second subtask of manipulation is to rotate the objects. *Ray-Casting* cannot rotate objects around arbitrary axes, and objects can only be rotated around the cast ray. *Go–Go*

Table 1 Selection techniques evaluated for mobile interaction, with respect to the proposed parameters

	Object distance/size	Density	Occlusion
Ray-Casting	−	++	++
	(Difficult to select small/distant objects)	(Easy to select objects in dense environments)	(Possible to select highly occluded objects)
Go–Go	−	−	+
	(Difficult to select small/distant objects)	(Difficult to select objects in dense environments)	(Can select highly occluded objects)
Aperture	++	−	−
	(Easy to select small/distant objects)	(Difficult to select in dense environments due to selection of multiple small/distant objects)	(Not possible to select highly occluded objects)
Occlusion	−	−	−
	(Difficult to select small/distant objects)	(Difficult to select in dense environments with small/distant objects due to finger size on display)	(Not possible to select highly occluded objects)

Table 2 Manipulation techniques evaluated for mobile interaction with respect to the proposed factors

	Ease of positioning	Ease of rotation	Precision
Z-Technique	+	−	+
	(Easy to position objects on screen locations but does not position objects off screen)	(No rotation)	(Easy to precisely position objects only on target locations visible in display)
Go–Go	++	++	−
	(Easy to position objects)	(Easy to rotate objects)	(Low precision due to the mapping of physical interaction)
Arc-Ball	−	++	+
	(No positioning)	(Easy to rotate objects)	(Easy to rotate objects with high precision but has average error rate due to combined DOF controls)
Ray-Casting	−	−	−
	(Restricted, no depth manipulation of object location)	(Hard to rotate objects on arbitrary axes. Rotation is restricted to ray axis)	(Low precision due to the mapping of physical interaction and lack of object depth manipulation)

can easily map the orientation of the user's hand to the object and rotate it around any arbitrary axis [2, 8, 9]. *Arc-Ball* is a preferable and precise virtual technique for rotating objects around any axis [2].

3.2.3 Precision

Object manipulation needs to be precisely performed to result in a minimum error rate. The physical interaction techniques *Go–Go* and *Ray-Casting* generally result in a high error rate due to inaccurate mapping of the user actions to the virtual environment. Virtual interaction techniques *arc-ball* and *Z-Technique* result in errors from non-separated degree-of-freedom (DOF) controls. The more DOFs are separated; the lower error rate is expected [21].

3.3 Navigation

Due to the fact that egocentric virtual environments are preferred for handheld devices, effective navigation is a high priority. Navigation techniques can be evaluated with respect to the following factors: distance, the number of rotations, cognitive load, and flexibility. Table 3 summarizes the well-known navigation techniques for comparing these factors in this study.

3.3.1 Distance

Travel distance is the most important attribute of the navigation task. For long distances, it is important to use a comfortable technique that will scale the input of the user and map it to the virtual environment. A virtual technique

Table 3 Navigation techniques evaluated with respect to the proposed parameters

	Distance	Number of rotations	Cognitive load	Flexibility
Pointing	− (Traveling long distances is hard for the user)	++ (Easy to rotate view physically)	++ (Low cognitive load)	++ (High flexibility, because user can change direction anytime)
Marking checkpoints	++ (Long distances are not a problem because the user will have an outer view of the environment and plan her route accordingly)	− (Does not provide real-time rotations)	− (High cognitive load)	− (Low flexibility, because once the path is marked, the user must switch to map mode from traveling mode to make any changes)

for scaling large movements is appropriate for this purpose [2]. *Pointing* is a physical technique that does not provide movement scaling in long distances: based on where the user points, the camera moves toward the specified direction. *Marking Checkpoints* is a virtual technique in which the user places markers in the map view on the ground and the camera moves visiting each of these points when map view mode is completed. This helps the user to travel long distances without effort [2].

3.3.2 Number of rotations

The travel path may require a large number of rotations to change the direction of movement. It is preferable to perform small tasks, such as rotating the view, physically. *Pointing* utilizes physical rotations and offers an effective solution to the user. *Marking Checkpoints* is a virtual route-planning technique, based on a map of the environment, and does not allow users to rotate the view directly [2].

3.3.3 Cognitive load

Interaction technique design must consider reducing the user's cognitive load [27]. During navigation, the user should be able to easily remember the route and actions taken over the long-term. *Pointing* offers real-time navigation so the user only needs to deal with short-term actions; therefore, she can easily focus on the route and the environment. *Marking Checkpoints* requires exploiting the user's long-term memory, which may prevent her from focusing on the environment.

3.3.4 Flexibility

During navigation, the user should be able to easily recover from mistakes; inflexible techniques increase the user's cognitive load. *Pointing* is a flexible technique that offers the user real-time feedback and a chance to undo or redo her actions. A route-planning technique such as *Marking Checkpoints* does not allow a user to easily modify her navigation path; it requires the user to switch to the exocentric view to revise the path, which makes it harder to recover from mistakes.

In this paper, we verify the actual performance of these existing methods on a mobile device with controlled experiments. These methods thus serve as baseline techniques for user-study comparisons with our proposed techniques, which we describe next.

4 Design objectives

Our main thesis is that *precise* selection of virtual objects, as well as their manipulation, and fluid navigation within the virtual world, are the most important aspects for interaction with virtual environments on mobile displays. Due to the physical constraints of the mobile device size and the constraints posed by the human fingers, direct manipulation on these displays suffers from limited precision, occlusion problems, and limitations to the size of the scene elements.

With this motivation, we first present a set of general *design objectives for mobile 3D interaction* with multi-touch input. Then, we inspect our proposed techniques in detail regarding design decisions made, metaphors chosen, and implementation details for the corresponding techniques.

4.1 Universal tasks

- *Precise selection and manipulation* The multi-touch selection technique should allow the user to perform precise selection of *small/distant* or *occluded* objects, as well as objects in *dense* environments. The manipulation technique should give importance to *ease of transformation*, *rotation*, and possibly *scaling*.

- *Ease of navigation* The navigation technique should be flexible and enable the user to easily travel long distances with comfort. The navigation technique should also offer ease of rotation, to facilitate travel and way finding tasks during navigation.
- *Egocentric view* Unlike exocentric (outside-in) approaches on tabletop 3D techniques, mobile 3D interaction techniques should focus on the egocentric view.
- *Connected feedback* Universal interaction techniques should provide appropriate feedback to the user, either visually or in another form. For example, throughout the manipulation, the user should experience constant visual/physical connection [19].

4.2 Mapping of input to 3D UI tasks

4.2.1 Bimanual and single-handed interaction

Multi-touch interaction techniques should allow bimanual interaction and two-finger interaction with one hand. For example, when the user interacts with a mobile device in a landscape orientation, both hands are generally required to hold the device. However, in certain cases, single-handed use (with multiple fingers of the dominant hand) would be beneficial, for example, while the user is holding a phone with the non-dominant hand (e.g. use in portrait mode).

4.2.2 Flexibility in reuse

Interaction techniques should be usable with other single-handed or physically based techniques. For example, it should be possible for the user to navigate in the scene with a single-touch-based technique or inertial trackers (e.g. a gyroscope) and select objects with a multi-touch technique.

4.2.3 Consistency

Consistent interface metaphors should be used when designing interaction techniques for the universal 3D UI tasks of navigation, selection, and manipulation.

4.2.4 High-level gestures

High-level gestures should be reserved for only low-level common tasks, such as for zooming in/out with the pinch gesture [19].

4.2.5 Degrees of freedom

Interaction techniques should target simultaneous rotation and translation, as well as rotation independence and DOF translation [1, 25].

4.3 Input modality

4.3.1 Constraints of mobile display

Interaction techniques should support the input modalities of commonly available mobile devices: that is, recognizing multi-touch input as a set of 2D contact points and the presence of low-precision inertial trackers (gyroscopes, accelerometers). Techniques should aim to solve the major interaction constraints of the mobile device: *finger occlusion*, *limited multi-touch input precision*, and *limited physical screen size*.

4.3.2 Presence of additional input methods

The techniques should not assume any additional sensor data than commonly available on mobile devices, for example, the availability of data for touch pressure or contact area for each finger, and hover input should not be assumed. However, with recent developments in this field [28], it should be possible to extend the proposed interaction techniques for possible common availability of these input modalities in the future.

4.3.3 Physical devices

Considering the mobile usage context, interaction methods should not assume the presence of additional physical tools (such as additional 3D pointing devices) to interact with the device.

5 Dual-finger 3D interaction

In this paper, we propose a set of dual-finger mobile 3D interaction techniques, illustrated in Fig. 2. These include two selection techniques: (1) *Dual-Finger Midpoint Ray-Casting* and (2) *Dual-Finger Offset Ray-Casting*; three techniques for separate object manipulation tasks: (3) *Dual-Finger Translation*, (4) *Dual-Finger Rotation*, and (5) *Dual-Finger Scale*; and one technique for navigation tasks: (6) *Dual-Finger Navigation*.

These techniques were inspired by the dual-finger 2D interaction technique proposed by Benko et al. [4] for precise selection of 2D UI widgets in desktop applications. While Benko et al. focus on solving precise selection task issues in 2D applications; we reformulate this input technique for universal 3D user interface tasks and formally study its suitability for 3D interaction.

5.1 Dual-Finger Midpoint Ray-Casting

The first selection technique, *Dual-Finger Midpoint Ray-Casting*, is illustrated in Fig. 3. The user employs two

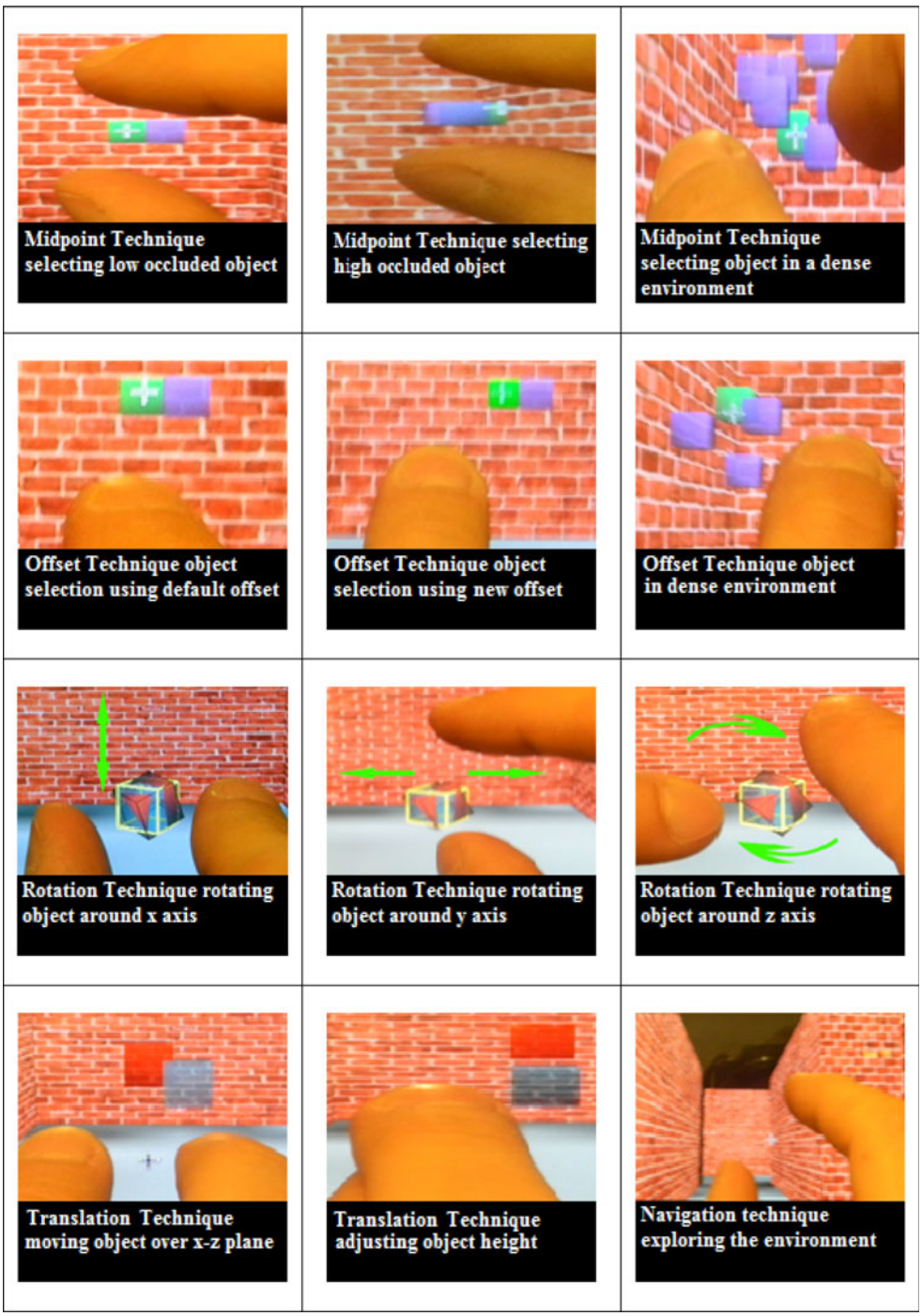

Fig. 2 *Dual-Finger 3D Interaction Techniques. First row* Dual-Finger Midpoint Ray-Casting Technique. *Second row* Dual-Finger Offset Ray-Casting Technique. *Third row* Dual-Finger Rotation Technique. *Fourth row* Dual-Finger Translation Technique and Dual-Finger Navigation Technique

fingers, *f1* and *f2*, for interaction. A crosshair marking the midpoint of these two fingers is drawn on location:

$$C = ((f1.x + f2.x)/2, (f1.y + f2.y)/2)$$

and a ray is generated from the center of projection toward the scene, which passes through the crosshair. To find the first object intersected by R, we perform a ray intersection test with each object in the scene. We highlight the intersected object by changing its color as a feedback to the user. Detailed explanation on *Ray-Casting* can be found in [29].

While the user has two contact points on the touch-screen, if she moves one of the fingers, this is transformed into a zoom centered at the crosshair location. For this purpose, we generate a ray from the center of projection, which passes through the crosshair location C to the environment and get a target point T, and direct the camera toward this point. Then, we apply a zoom by modifying the projection matrix, in a similar effect to the two-finger pinch gesture used for zooming in 2D interaction on smartphones. While there is a highlighted object, if the user

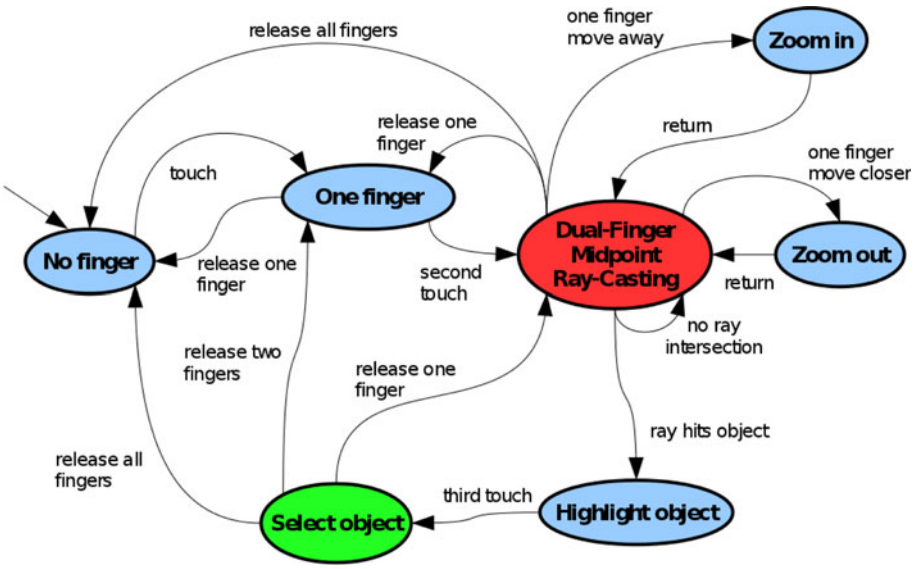

Fig. 3 State diagram for Dual-Finger Midpoint Ray-Casting technique

performs any third touch action, the object is selected and highlighted with a different color as a feedback.

5.2 Dual-Finger Offset Ray-Casting

The second selection technique *Dual-Finger Offset Ray-Casting* is illustrated in Fig. 4. In this technique, only one finger is used as a pointer in the 3D environment. A crosshair follows the finger with an offset o, and its position is calculated as:

$$C = (f1 \cdot x + o \cdot x, f1 \cdot y + o \cdot y)$$

which is the finger position with the amount of offset added to it. Similar to the *Dual-Finger Midpoint Ray-Casting* method, we construct a ray R and find the first intersected object with the minimum distance.

When the user places a second finger *f2* on the touch-screen, there are two possible interpretations of this input. To determine the mapping, the distance d between *f1* and *f2* touch points is computed:

$$d = \sqrt{(f1x - f2x)^2 + (f1y - f2y)^2}$$

and if d is larger than a threshold t, we reposition the crosshair to the midpoint between the fingers *f1* and *f2*, as in the midpoint technique. If both fingers move, then a zoom is performed centered at the crosshair location. For this purpose, we project the crosshair location C to the environment to get a target point T, which we direct the camera toward. Then, we modify the projection matrix by adding the zoom effect. By default, the crosshair is above the finger; selecting objects that are close to lower border of the screen is difficult; thus, the user should place *f2* below *f1* to offset the crosshair below the finger.

In the second case, if d is less than t and there is a highlighted object O, then the user selects the object. The object color is highlighted differently as a feedback for the user.

5.3 Dual-Finger Midpoint Translation

The first manipulation technique is named *Dual-Finger Translation* and illustrated in Fig. 5. It is assumed that the user already selected the object with two fingers, *f1* and *f2,* as described above, and the two fingers are currently touching the display before starting manipulation. There are two alternative interpretations of the user's input. If the distance d between *f1* and *f2* is less than a threshold t, then it is assumed that the fingers are adjoint. For all the experiments in this paper, we have empirically used 100 pixels for the threshold t, as an estimated distance between the tips of two adjoint fingers on the screen. To translate the selected object on y axis (vertical to view plane), both fingers are moved up or down; thus, the y component of selected object O is updated accordingly. If d is larger than threshold t, the fingers are thought to be split; therefore, the active subtask is to position the object on the x–z plane where the horizontal ground surface of the environment lies. The crosshair position C is projected from the view plane to the 3D environment ground surface to get point E on x–z plane; then x and z components for location L of selected object O are calculated as $Lx = Ex$, $Lz = Ez$, and Ly remains unmodified. This three degree-of-freedom (DOF) positioning technique is decomposed into two integrated DOF and one separate DOF for two separate positioning subtasks described. For translating the objects to points that are not currently in the view, a

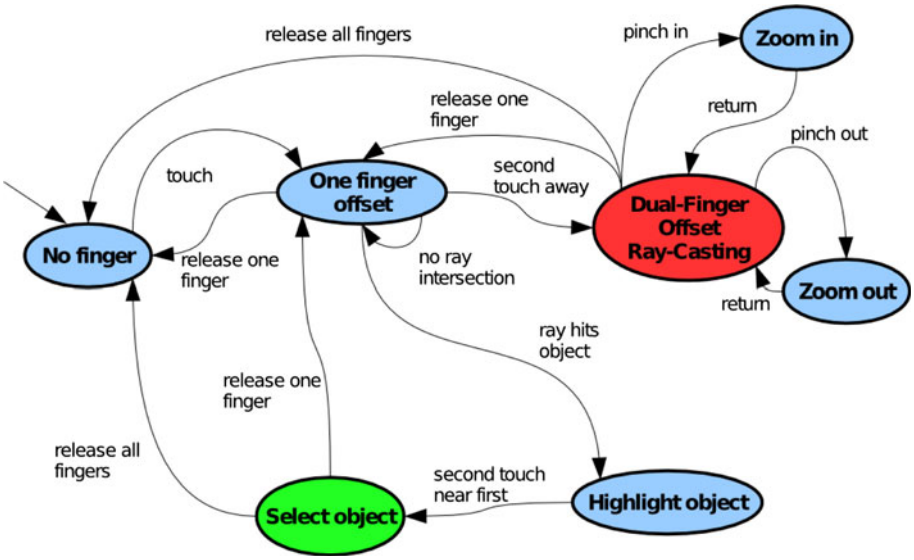

Fig. 4 State diagram for Dual-Finger Offset Ray-Casting technique

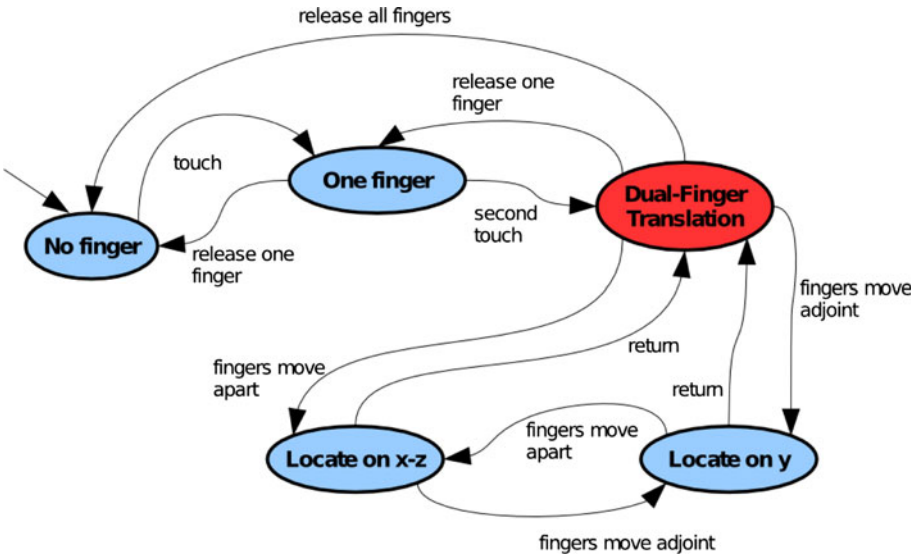

Fig. 5 State diagram for Dual-Finger Translation technique

semi-automated method is used. When the user moves fingers to the edge or corner of the screen, she starts to rotate the camera toward the direction of the pushed edge or corner.

5.4 Dual-Finger Rotation

The *Dual-Finger* Rotation technique is illustrated in Fig. 6. The user employs two fingers *f1*, *f2* to rotate the object along *x*, *y*, and *z* axes. When she moves both fingers parallel to *x* axis in the same direction, the object is correspondingly rotated around the *y* axis. The same applies to moving the fingers parallel to *y* axis in the same direction to rotate the object around *x* axis. Rotation around z axis is performed by a twisting action by moving the fingers parallel to x axis or y axis, in the opposite direction.

5.5 Dual-Finger Scaling

The *Dual-Finger Scaling* interaction technique (Fig. 7) is a natural extension of these techniques. This technique allows the user to perform pinch gestures vertically to scale the object along the *y* axis and horizontally to scale object along the *x* axis. If the user moves two fingers adjointly, vertically upwards or downwards, the object is scaled along the *z* axis.

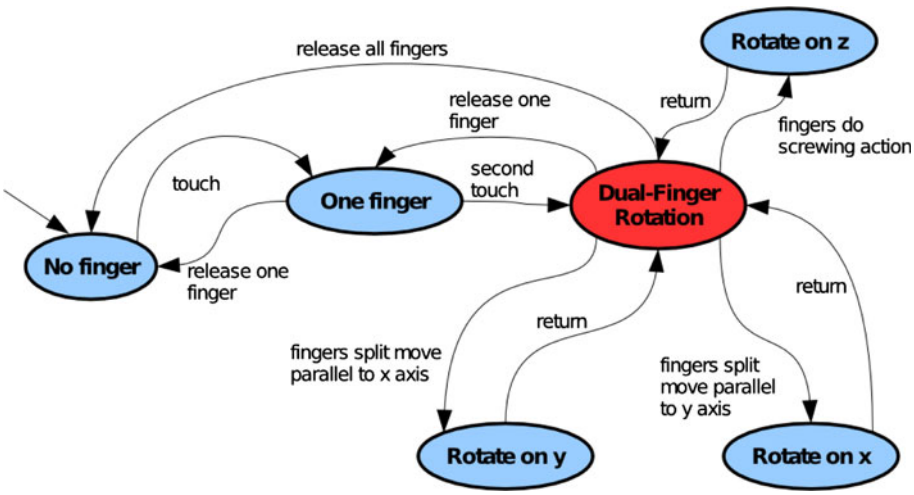

Fig. 6 State diagram for Dual-Finger Rotation technique

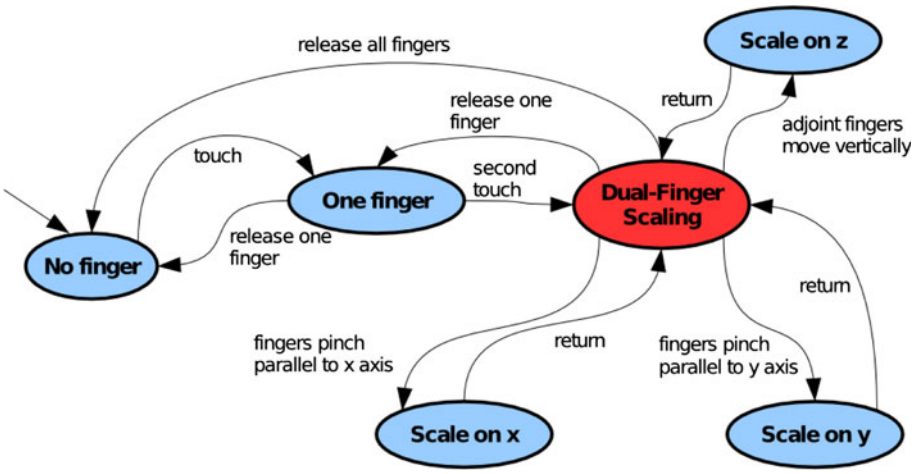

Fig. 7 State diagram for Dual-Finger Scaling technique

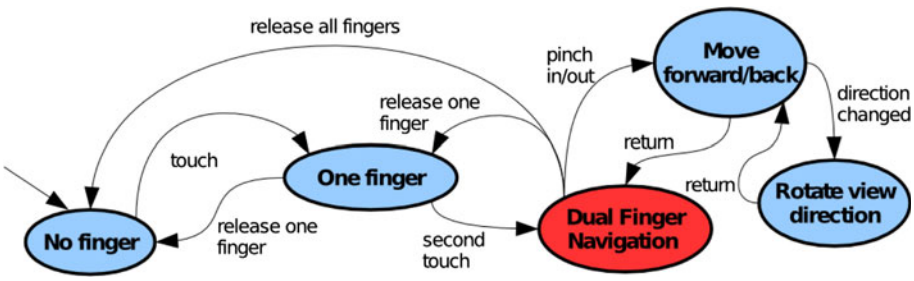

Fig. 8 State diagram for Dual-Finger Navigation technique

5.6 Dual-Finger Navigation

The proposed navigation technique, *Dual-Finger Navigation*, again requires the use of two fingers *f1* and *f2*. This method is illustrated in Fig. 8. The user performs standard *pinch-in* gesture to move forwards and *pinch-out* gesture to move backwards on the *x*–*z* plane. Traveling in vertical *y* axis is avoided and omitted for more realistic navigation. The midpoint of the two fingers is again marked with a crosshair to specify the direction to move. While moving with pinch gestures, changes in the midpoint yield a view direction change.

These techniques can flexibly be used together with other single-handed or physically based techniques, or combined together. For example, an application may support the user's navigation in the scene with a single-touch-based technique or inertial trackers (e.g. gyroscope) and perform selection with the dual-finger technique, while

another application may support dual-finger navigation and single-touch tapping for object selection.

6 Controlled experiment

6.1 Goals

The main objective of this test is to evaluate the proposed *dual-finger interaction set*. The experiment design is based on the following hypotheses:

H1. *Dual-Finger Midpoint Ray-Casting and Dual-Finger Offset Ray-Casting selection techniques are faster and more precise than Image-Plane Technique Tapping, physical Ray-Casting, and Go–Go techniques.* Because the user touches with her fingers during *Tapping*, finger size is a problem when selecting small, occluded objects or objects in dense environments. *Ray-Casting* and *Go–Go* will take longer time during selections of small objects because small movements due to hand shaking may have a more profound effect in the virtual environment. Therefore, it is hypothesized that *Dual-Finger Midpoint Ray-Casting* and *Dual-Finger Offset Ray-Casting*, which are less affected these limitations, are faster.

H2. *Dual-Finger Translation* manipulation technique is more accurate and faster than *Go–Go* and *Z-Technique*. Since the proposed translation technique is based on DOF separation, users' actions will be more coordinated, and they will spend less time in error correction. By comparing *Dual-Finger Translation* and *Z-Technique*, we measure the performances of DOF separation as x–z, y against x–y, z. Translating the object easily on the horizontal space will give higher degree of depth cues to the user and allow her to adjust object height separately. Therefore, *Dual-Finger Translation* should exhibit higher performance in both interaction time and reduced error rate.

H3. *Dual-Finger Rotation* is a more accurate and faster rotation technique than *Arc-Ball* and *Go–Go* techniques. Since the proposed rotation technique is based on DOF separation, users will be more coordinated and will spend less time in error correction. Thus, *Dual-Finger Rotation* should have higher performance in timing and reduced error rate.

H4. *Dual-finger navigation* is a faster and more comfortable navigation technique to the user than *Pointing* and *Marking Checkpoints*. With the *pointing* technique, users have to physically perform rotations. Going backwards requires users to perform a 180° physical rotation. Constantly changing direction takes significant amount of time. In the map-based *Marking Checkpoints*, the user frequently needs to open the map and plan the route. Therefore, *Dual-Finger Navigation* should be faster.

6.2 Apparatus

The experiment was conducted on an iPhone 4 [30] with the iOS 4.3.5 operating system. This mobile device has a screen resolution of 960 × 640 pixels (326 PPI) and a 3.5″ diagonal length. Test applications were implemented using the cocos3d graphics engine framework [31]. Tests were performed while the mobile device was connected to a MacBook Pro 13″, and outputs of the tests, such as task completion time, error rate etc., were displayed on the Xcode 3.2.6 console with iOS 4.3 SDK [32].

6.3 Implementation of techniques in comparison

The first well-known selection technique for comparison, *Tapping,* was implemented as a virtual technique where the participants tapped on the target object to select it. The second selection technique, *Ray-Casting*, was implemented as a hybrid technique where the participants pointed the ray physically using the device's gyroscope sensor to the target object to highlight it, then touched the screen once to confirm selection. The last selection technique, *Go–Go*, was also implemented as a hybrid technique where the participants pointed the virtual hand physically similar to *Ray-Casting*; touched the screen and performed swipe up and down gestures to adjust the arm length; and placed two fingers to select the object that intersected with the virtual hand.

The first positioning technique for comparison, *Z-Technique*, was implemented as a virtual technique where the participants moved their finger up, down, left and right on the screen, to position the object on the x–y plane, and moved two fingers up and down to adjust the depth of the object along the z axis. To complete the positioning task, they placed three fingers on the screen. The second positioning technique, *Go–Go*, was implemented similar to the selection technique. The selected object followed the hand just below it; and when the participants wanted to complete the positioning task, they were asked to place two fingers on the screen.

The rotation technique *Arc-Ball* was implemented as a virtual technique where the participants could drag the object to any direction to roll it toward and placed two fingers to complete the task. The second rotation technique, *Go–Go*, was implemented as a hybrid technique where the participants tilted the device around the x, y and z axes to rotate the selected object and touched on the screen to complete the task.

The first navigation technique *Pointing* was implemented as a hybrid technique, which used the gyroscope to

perform view point rotations, and the screen interactions to perform movement toward the specified camera direction. The second technique *Marking Checkpoints* was implemented to allow the participants to switch to the map mode, which presented a view point on top of the scene. The participants placed two fingers to switch to the exocentric view and placed checkpoints on the scene to plan the route, then placed two fingers on the screen user again to exit from the map mode and start moving through marked checkpoints. While moving, the participants were allowed to look around using the gyroscope sensor.

6.4 Participants

We performed this set of experiments on fifteen participants (three females and twelve males) with varying levels of mobile experience. There were thirteen users of a smartphone with touch-screen and two users of a mobile device with keyboard and non-touch displays. There were five novice users, seven users with average experience, and three experts with significant gaming experience. Following Apple's Human Interface Guidelines, among the male and female participants, we assume an average of 1 cm^2 (44 $\times$ 44 pixels) finger size on the screen [33] and do not consider the finger size to be a blocking factor for the experiment.

6.5 Design

For all tests, we used a repeated measures design. For each interaction technique, the participants had 10 min of training period before the tests. Furthermore, before each task, a button appears on the screen, when the participant feels ready, she presses the button, and a 3 s countdown starts to prepare the participants. For each participant, the complete test lasted approximately 60 min, divided into three blocks of approximately 20 min, separated by a 3 min break.

6.5.1 Object selection task

Participants performed selection using *Dual-Finger Midpoint Ray-Casting*, *Dual-Finger Offset Ray-Casting*, *Ray-Casting*, *Go–Go*, and *Tapping* techniques. A yellow colored box was placed in the environment, and participants were asked to select it under three different conditions. In the first case, we measured the object size and distance effect: in each trial, the object was placed with higher depth, and the area of the object on the screen was reduced. In the second case, the occlusion effect on object selection task was measured: a secondary object occluded a target yellow cube with different levels. In the final set of trials, the object density of the environment increased at each trial to measure the object density effect on the performance of selection task. Participants were asked to select the target objects as quickly as possible.

In this task, the independent variables are TECHNIQUE, ENVIRONMENT PARAMETERS, and TASK DIFFICULTY. There are five levels of TECHNIQUE: *Dual-Finger Midpoint Ray-Casting*, *Dual-Finger Offset Ray-Casting*, *Tapping*, *Ray-Casting* and *Go–Go*. The presentation order of TECHNIQUE was counterbalanced across participants. The techniques were presented to the participants for varying ENVIRONMENT PARAMETERS: *object size*, *object occlusion*, and *environment density*. TASK DIFFICULTY for the first environment parameter varies from 0.25 to 0.01 cm^2, for the second type of environments difficulty varies between 10 and 95 % occlusion level, and lastly, for dense environments difficulty varies between 1 and 12 additional objects in the scene. Each combination of these variables was tested on 15 participants. Therefore, in total, the design of the experiment resulted in:

15 Participants $\times$ TECHNIQUE
$\times$ ENVIRONMENT PARAMETERS
$\times$ TASK DIFFICULTY
$=$ 4500 total trials.

6.5.2 Object positioning task

Object positioning was performed through *Dual-Finger Translation*, *Go–Go*, and *Z-Technique* techniques. A red-colored box was placed in the environment, and the participants were asked to place it into an equally sized container box which was transparent [34] and cyan colored. The participants were asked to position the target objects into place as quickly as possible.

Thus, we have measured positioning task completion data for three positioning techniques, where each data block included a positioning time, a horizontal error rate, and a vertical error rate. The error rates were calculated using the following formula [5]:

$$E_h = \frac{\sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}}{D_s} \times 100\,\%$$

$$E_v = \frac{\sqrt{(y_0 - y_1)^2 + (z_0 - z_1)^2}}{H_s} \times 100\,\%$$

where E_h and E_v represent horizontal and vertical error rates of object positioning in the target container, respectively. Variables x_1, y_1, z_1 and x_0, y_0, z_0 are the geometric positions of the container and selected objects; D_s is the horizontal diagonal of the box; and H_s is height of the box.

In this task, the independent variables are TECHNIQUE and DISTANCE. There are three levels of TECHNIQUE: *Dual-Finger Translation*, *Z-Technique*, and *Go–Go*.

DISTANCE represents the distance between the object to be positioned and target object location and varies between 1.8 and 4.5 units in the 3D environment. Each combination of these variables was tested on 15 participants. Thus, the design of the experiment resulted in:

15 Participants × TECHNIQUE × DISTANCE
= 900 total trials.

6.5.3 Object rotation task

Rotating the selected object was performed through *Dual-Finger Rotation*, *Go–Go*, and *Arc-Ball* techniques. A red box was placed in the environment, and another transparent, cyan colored and equally sized container box was placed in the same location. This container box was rotated around a single axis in the first tests, around two axes for medium difficulty tests, and around some arbitrary axis for the difficult tests. The participants were asked to rotate the red box until they think the box fits into the container box.

Thus, we have measured rotating task completion data in total for three rotation techniques, where each data block includes a rotation time, and three error rates of rotation around each axis. Since the rotated object is symmetric and can rotate with additional 180° and still be aligned with the target, rotation of the object around one axis is calculated as rotation of the target object twice in a 360 degree circle. Thus, error rates are calculated for each axis separately, using the following formula:

$$\Delta_{angle} = \frac{\left|(C_{angle} \bmod 180) - (O_{angle} \bmod 180)\right|}{180} \times 100\,\%$$

The independent variables are TECHNIQUE and ROTATIONAL COMPLEXITY. There are three levels of TECHNIQUE: *Dual-Finger Rotation*, *arc-ball*, and *Go–Go*. ROTATIONAL COMPLEXITY varied between one- and three-axis rotation. For the one-axis task, rotations are constrained to take place only around the z axis, pointing toward the participant. While rotating around two axes, the rotations are only allowed around the y–z and y–x axes.

Each combination of these variables was tested on 15 participants. Thus, in total, the design of the experiment resulted in:

15 Participants × TECHNIQUE
× ROTATION COMPLEXITY
= 900 total trials.

6.5.4 Navigation task

Participants navigated through the map (Fig. 9) using *Dual-Finger Navigation*, *Pointing* and WIM-based *Marking Checkpoints* techniques for 5 tasks. In the initial task, the participants were asked to visit Room 1, in the second task visit Room 2 and in the third task to Room 3, with increasing distances. For more challenging test cases, the participants were asked to visit both Room 1 and Room 2 in the fourth task; and all the rooms in the final task. The purpose of this design was to increase the length of the path and the number of rotations performed so that we could measure these effects on the methods tested.

The independent variables are TECHNIQUE and DISTANCE. There are three levels of TECHNIQUE: *Dual-Finger Navigation*, *Pointing*, and *Marking Checkpoints*. DISTANCE is a measure of the length of the path taken, divided into five levels, and represents task difficulty. Each combination of these variables was tested on 15 participants. Therefore, the design of the experiment resulted in:

15 Participants × TECHNIQUE × DISTANCE
= 225 total trials.

7 Results

7.1 Object selection

7.1.1 Object size

The repeated measures analysis of variance (ANOVA) on experimental results found a significant effect for TECHNIQUE (F 1,14 = 525.51, $p < 0.001$) on selection time of small objects. A pairwise comparison revealed significant differences between *Dual-Finger Midpoint Ray-Casting* (mean: 1.9 s) and *Dual-Finger Offset Ray-Casting* (mean: 3.1 s) ($p < 0.001$). Further pairwise comparisons showed significant differences ($p < 0.001$) between *Dual-Finger Midpoint Ray-Casting* and the three other methods: *Tapping* (5.7 s); *Ray-Casting* (3.6 s); *Go–Go* technique (13.6 s) (Fig. 10). Furthermore, a pairwise comparison between the standard *Ray-Casting* and *Tapping* methods revealed a significant difference ($p = 0.003$), suggesting that the standard *Ray-Casting* technique is more viable than *Tapping* for selection of small objects on mobile devices. Interaction of TECHNIQUE and TASK DIFFICULTY (i.e. object size) has a noteworthy effect; one possible reason is that *Go–Go* and *Tapping* methods' performance is less effective on smaller target objects, while no such interactions are observed for the proposed Dual-Finger selection techniques and the *Ray-Casting* technique. During the experiments, there were 20 task difficulty levels, and adjacent difficulty levels do not indicate a high variation of selection time results. There was a learning effect for the last trials of the test, and due to this effect, it is possible to observe a slight decrease in mean selection task completion times for the last object in Fig. 10, though this decrease is not significant.

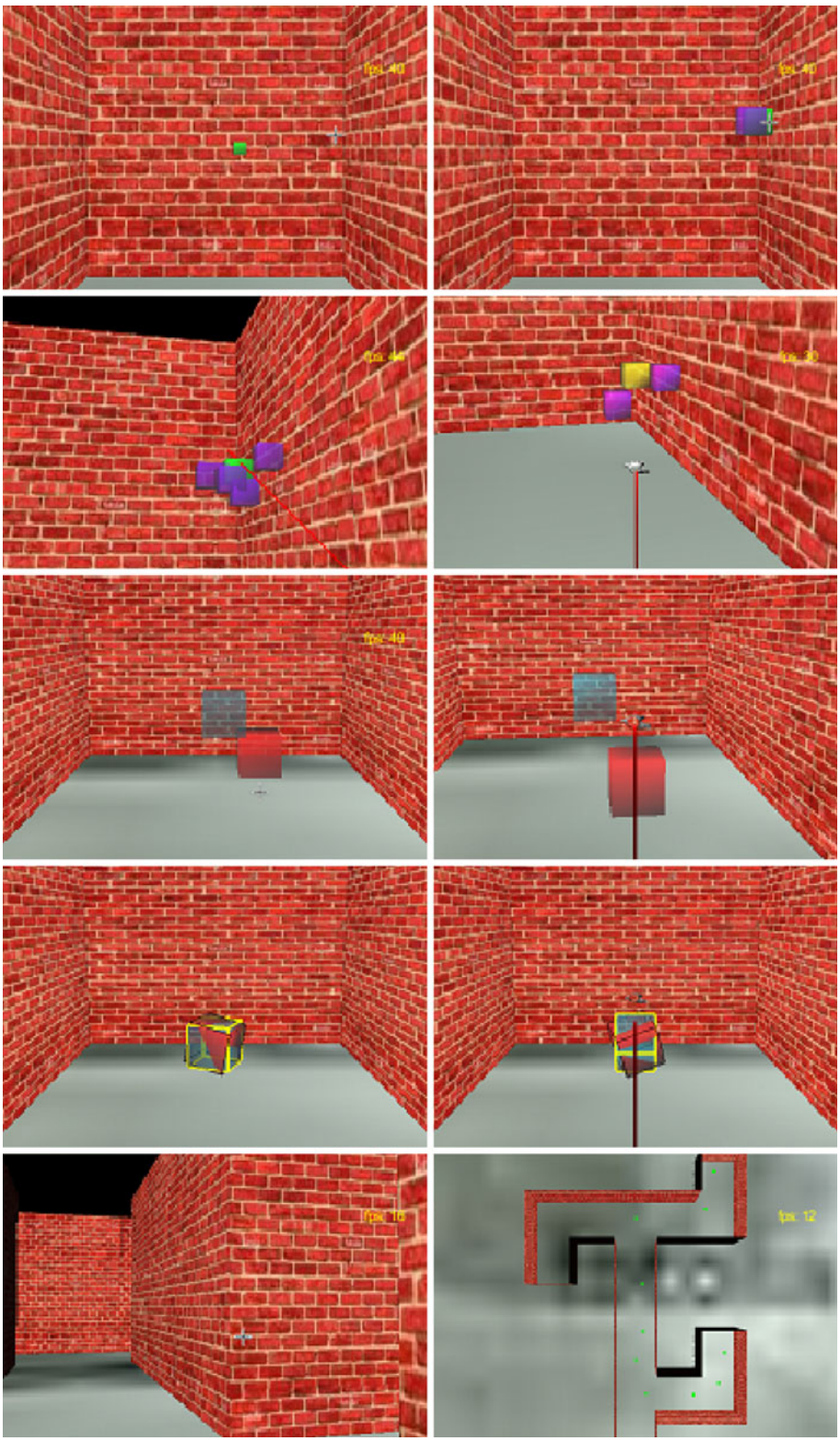

Fig. 9 Screen captures from various test scenes, from *left* to *right* and *top* to *bottom*: *Dual-Finger Midpoint Ray-Casting* selecting a small object, *Dual-Finger Offset Ray-Casting* selecting an occluded object, *Ray-Casting* selecting an object from a dense environment *Go–Go* technique selecting an object from a dense environment, *Dual-Finger Translation* positioning an object on the x–z plane, *Go–Go* technique positioning an object, *Dual-Finger Rotation* and *Go–Go* techniques rotating an object, *Dual-Finger Navigation* moving and *Marking Checkpoints* technique planning a path in the environment

7.1.2 Object occlusion

The ANOVA found a significant effect for TECHNIQUE also in selecting occluded targets ($F_{1,14} = 1{,}019.667$, $p < 0.001$). A pairwise comparison revealed no statistically significant difference between *Dual-Finger Midpoint Ray-Casting* (2.7 s) and *Dual-Finger Offset Ray-Casting* (2.6 s) ($p = 0.509$). Further pairwise comparisons showed significant differences ($p < 0.001$) between *Dual-Finger Midpoint Ray-Casting* and the three other methods: *Tapping*

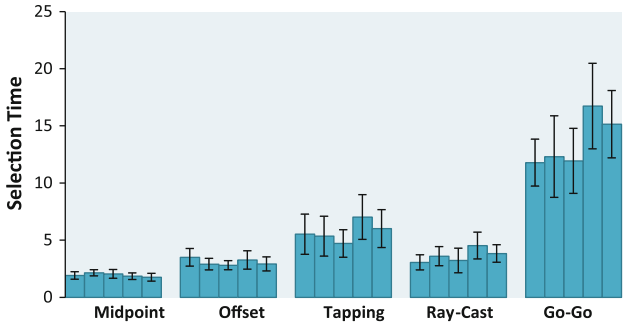

Fig. 10 Mean selection time for each technique under different levels of target object size. The *bars* for each technique represent the target size for 0.25, 0.2, 0.08, 0.04 and 0.01 cm^2, respectively. *Error bars* represent a 95 % confidence interval

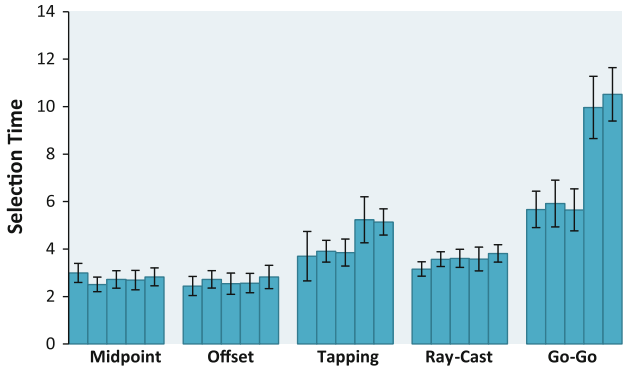

Fig. 11 Mean selection time for each technique under different levels of occlusion. The *bars* for each technique represent the target object's occlusion level as 10, 30, 50, 70, and 95 %, respectively. *Error bars* represent a 95 % confidence interval

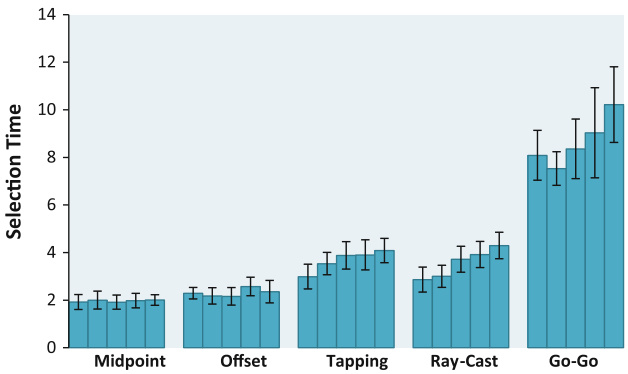

Fig. 12 Mean selection time for each technique under different levels of environment density. The *bars* for each technique represent environment density as 1, 5, 7, 10, and 12 objects in environment respectively. *Error bars* represent a 95 % confidence interval

(4 s); *Ray-Casting* (3.5 s); *Go–Go* technique (7.5 s) (Fig. 11). A pairwise comparison between the standard *Ray-Casting* and *Tapping* methods revealed a significant difference ($p = 0.012$), suggesting that the *Ray-Casting* method is more effective than *Tapping* for selection of partially occluded objects on mobile devices. There is a significant interaction between TECHNIQUE and TASK DIFFICULTY (i.e. occlusion level). It may be due to the fact that the *Go–Go* and *Tapping* techniques' selection performance is inferior on highly occluded target objects, while no such interaction was observed with the proposed Dual-Finger selection techniques and *Ray-Casting*.

7.1.3 Environment density

The ANOVA found a significant effect for TECHNIQUE on selection time inside dense environments (F 1,14 = 1300.024, $p < 0.001$). A pairwise comparison revealed no statistically significant differences between *Dual-Finger Midpoint Ray-Casting* (2 s) and *Dual-Finger Offset Ray-Casting* (2.3 s) ($p = 0.116$). Further pairwise comparisons showed significant differences ($p < 0.001$) between *Dual-Finger Midpoint Ray-Casting* and the three other methods: *Tapping* (3.7 s); *Ray-Casting* (3.6 s); *Go–Go* technique (8.6 s) (Fig. 12). However, pairwise comparisons between the standard *Ray-Casting* and *Tapping* revealed no significant difference ($p = 0.458$), suggesting that Ray-Casting is not more precise compared to image-plane tapping in dense environments. Interaction of TECHNIQUE and TASK DIFFICULTY (level of density) has a noteworthy effect on selection time in dense environments; a possible explanation is that *Ray-Casting, Go–Go,* and *Tapping* methods perform less effectively in dense environments, while no such interactions were observed for the Dual-Finger techniques.

These results support **H1** that *Dual-Finger Midpoint Ray-Casting* and *Dual-Finger Offset Ray-Casting* selection techniques are faster and more precise than the image-plane technique *Tapping*, physical *Ray-Casting*, and *Go–Go* techniques. While the *Dual-Finger Midpoint Ray-Casting* and *Dual-Finger Offset Ray-Casting* techniques yield similar task completion times for the same density and occlusion levels in a scene, the midpoint method provides a better performance with smaller objects.

7.2 Object manipulation

7.2.1 Object positioning

7.2.1.1 Positioning time The repeated measures analysis of variance (ANOVA) on experimental results found a significant effect for TECHNIQUE *(*F 1,14 = 4049.940, $p < 0.001$*)* on object positioning task completion time. Pairwise comparison showed no significant task completion time difference between *Dual-Finger Translation* (5.4) and *Z-Technique* (5.5 s) ($p = 0.578$) but a significant difference between *Dual-Finger Translation* and *Go–Go*

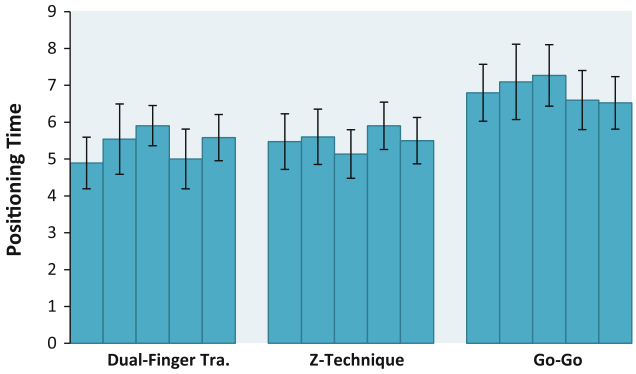

Fig. 13 Mean object positioning time for each technique under different levels of task complexity. The *bars* for each technique represent task complexity, with the distance between the selected object and the target container as 1.8, 2.3, 2.9, 3.1, and 4.5 units in the 3D scene, respectively. *Error bars* represent a 95 % confidence interval

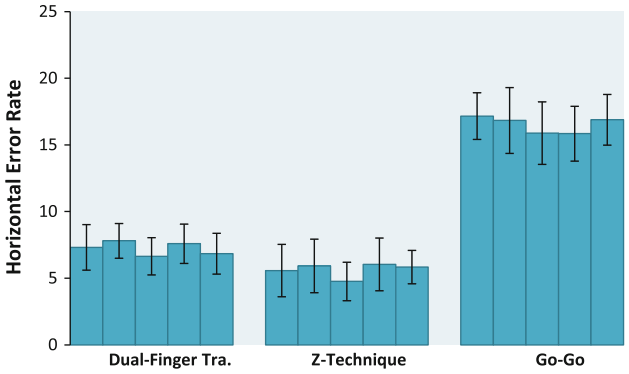

Fig. 14 Mean object positioning error's horizontal component for each technique under different levels of task complexity. The *bars* for each technique represent task complexity, the distance between the selected object and the target container as 1.8 units, 2.3 units, 2.9 units, 3.1 units, and 4.5 units in the 3D scene, respectively. *Error bars* represent a 95 % confidence interval

(6.9 s) ($p < 0.001$) (Fig. 13). Furthermore, a pairwise comparison between *Z-Technique* and *Go–Go* yielded a significant difference ($p < 0.001$, two-tailed, $t(14) = -5.558$), which reveals *Z-Technique* to be a faster object positioning technique on mobile devices.

7.2.1.2 Horizontal positioning error The ANOVA found a significant effect for TECHNIQUE on the horizontal object positioning error rate ($F_{1,14} = 11{,}250.138$, $p < 0.001$). Pairwise comparisons showed no significant difference in error rate between *Dual-Finger Translation* (7.2 %) and *Z-Technique* (5.6 %) ($p = 0.56$) but a significant difference between *Dual-Finger Translation* and *Go–Go* (16.5 %) ($p < 0.001$) (Fig. 14). Furthermore, a pairwise comparison between *Z-Technique* and *Go–Go* showed a significant difference ($p < 0.001$), which offers *Z-Technique* to be a more horizontally accurate object positioning technique in the mobile context.

7.2.1.3 Vertical positioning error The ANOVA found a significant effect for TECHNIQUE also on the vertical object positioning error rate ($F_{1,14} = 1{,}738.266$, $p < 0.001$). Pairwise comparisons showed a significant task completion vertical error rate difference between *Dual-Finger Translation* (3.2 %) and *Z-Technique* (9 %) ($p < 0.001$) and *Go–Go* (6.7 %) ($p < 0.001$) (Fig. 15). Pairwise comparison of *Z-Technique* and *Go–Go* showed a significant difference ($p < 0.001$), which reveals *Z-Technique* to be a more vertically accurate object positioning technique in the mobile context.

Interaction of TECHNIQUE and DISTANCE has a noteworthy effect on object positioning task completion time, and the horizontal and vertical error rates. One possible explanation of this interaction is the dependency of

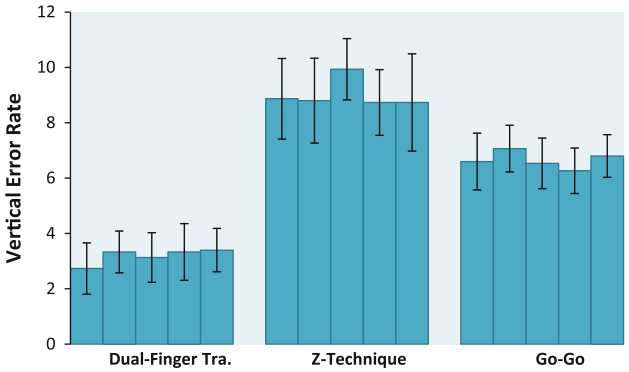

Fig. 15 Mean object positioning error's vertical component for each technique under different levels of task complexity. The bars for each technique represent task complexity, the distance between the selected object and the target container as 1.8 units, 2.3 units, 2.9 units, 3.1 units, and 4.5 units, respectively. *Error bars* represent a 95 % confidence interval

the *Go–Go* method's performance on distance, while other techniques do not demonstrate such dependency.

These results partially support **H2** that *Dual-Finger Translation* is a faster and more precise than *Z-Technique* and *Go–Go* techniques. Although *Dual-Finger Translation* provides a more precise solution than the other techniques, the *Z-Technique* provides a similar time performance for object positioning.

7.2.2 Object rotation

7.2.2.1 Rotation time The repeated measures analysis of variance (ANOVA) on experimental results found a significant effect for TECHNIQUE on task completion time ($F_{1,14} = 1{,}223.363$, $p < 0.001$). Pairwise comparisons showed no significant task completion time difference

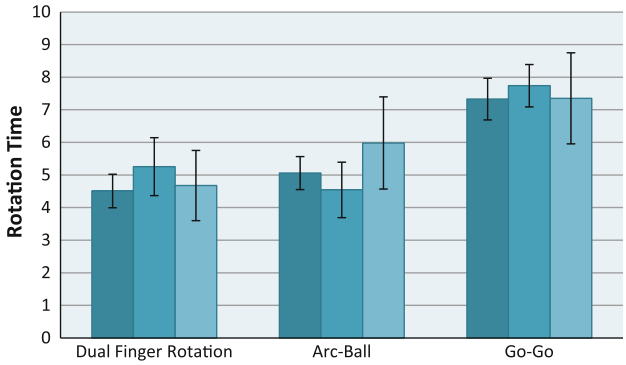

Fig. 16 Mean object rotation time for each technique under different levels of task complexity. The *bars* for each technique represent task complexity and number of rotation axes as one, two, and three. *Error bars* represent a 95 % confidence interval

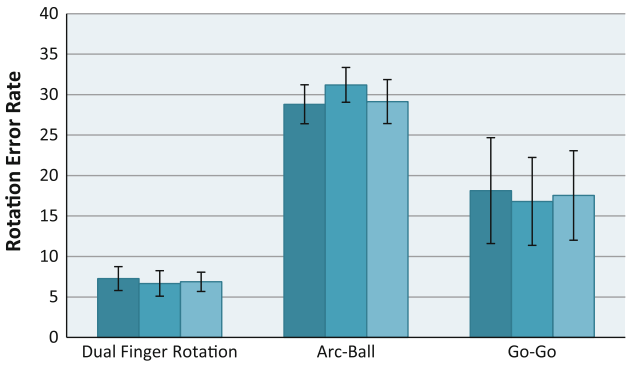

Fig. 17 Mean object rotation error rate for each technique under different levels of task complexity. The *bars* for each technique represent task complexity and number of rotation axes as one, two, and three. *Error bars* represent a 95 % confidence interval

Dual-Finger Rotation (4.9 s) versus *Arc-Ball* (5.2 s) ($p = 0.257$), but a significant difference with *Go–Go* (7.7 s) ($p < 0.001$) (Fig. 16). Furthermore, a pairwise comparison of *Arc-Ball* and *Go–Go* showed a significant difference ($p < 0.001$), which reveals *Arc-Ball* to be a faster object rotation technique.

7.2.2.2 Rotation error The ANOVA found a significant effect for the object rotation technique on the cumulative error rate (F 1,14 = 1410.097, $p < 0.001$). Pairwise comparisons showed a significant cumulative error rate difference between *Dual-Finger Rotation* (7.2 %), *Arc-Ball* (29.5 %) ($p < 0.001$), and *Go–Go* (17 %) ($p < 0.001$) (Fig. 17). Pairwise comparison of *Arc-Ball* and *Go–Go* resulted in ($p < 0.001$), which reveals *Go–Go* to be a more accurate object rotation technique.

These results partially support **H3** that *Dual-Finger Rotation* is faster and more precise than *Arc-Ball* and physical *Go–Go*. The rotation task completion time results show that *Dual-Finger Rotation* provides a similar performance as *Arc-Ball*; however, within time constraints, it provides a more precise rotation solution.

7.3 Navigation

The ANOVA found a significant effect for TECHNIQUE on task completion time (F 1,14 = 17935.919, $p < 0.001$). Pairwise comparisons showed no significant time difference between *Dual-Finger Navigation* (13 s) and *Pointing* (13.6 s) ($p = 0.253$); however, showed a significant difference between *Dual-Finger Navigation* and map-based *Marking Checkpoints* (26.9 s) ($p < 0.001$) (Fig. 18). Pairwise comparison of *Pointing* and *Marking Checkpoints* showed a significant difference ($p < 0.001$), which reveals *Pointing* to be a more effective and faster navigation technique.

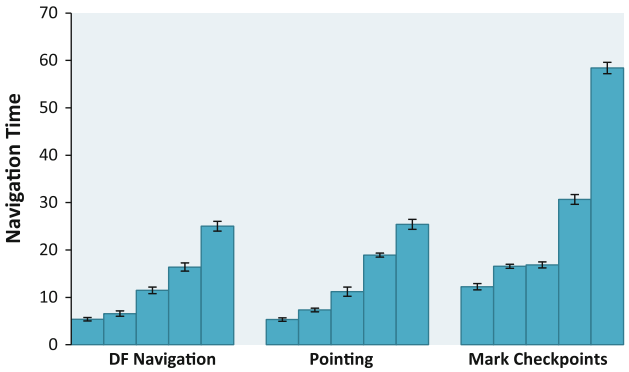

Fig. 18 Mean navigation task completion time for each technique under different levels of task complexity. The *bars* for each technique represent task complexity. *Error bars* represent a 95 % confidence interval

These results partially support **H4.** *Dual-Finger Navigation* is faster than *Marking Checkpoints* but has similar task completion time to *Pointing*. However, the following subjective user evaluation revealed that our proposed technique is perceived as easier to use than both physical *Pointing* and *Marking Checkpoints*.

7.4 Subjective evaluation

While evaluating our designs, we asked participants to fill questionnaires about their impression of the presented techniques under comparison during test. In the forms, we asked them to grade the selection, manipulation, and navigation techniques by grading their ease of use and familiarity between 1 and 7 according to how they felt. The results show that, after a short training session, participants were comfortable and capable of performing the tasks, using the techniques proposed. Questionnaire responses confirmed that selection techniques *Dual-Finger Midpoint Ray-Casting* and *Dual-Finger Offset Ray-Casting* were

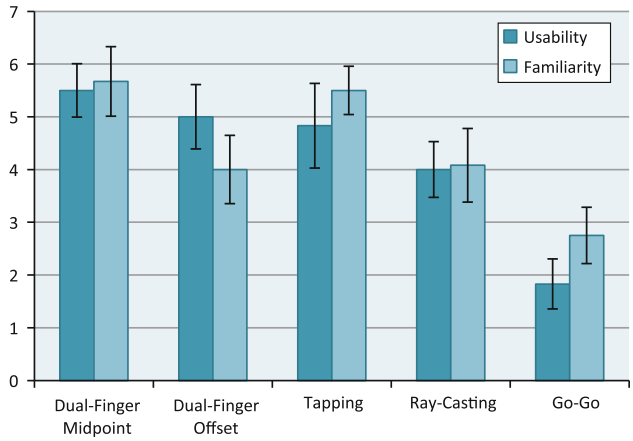

Fig. 19 Subjective evaluation of object selection techniques. *Error bars* represent 95 % confidence interval

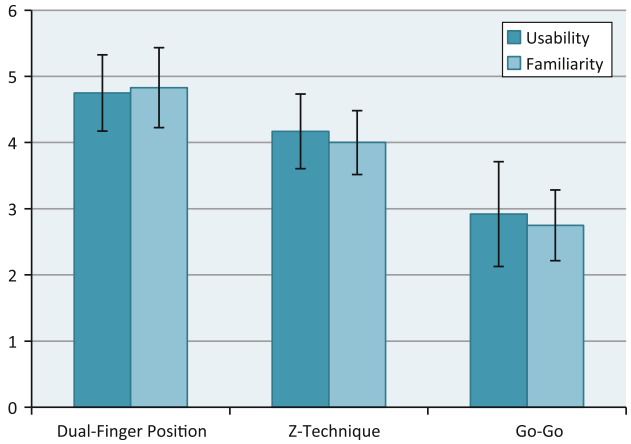

Fig. 20 Subjective evaluation of object positioning techniques. *Error bars* represent 95 % confidence interval

usable and easy to learn (5.5/7 for midpoint and 5/7 for offset). However, these techniques differed in familiarity as evaluated by the participants (5.67/7 for midpoint and 4/7 for offset). Questionnaire results also showed that object translation technique *Dual-Finger Translation* was marginally easy to learn (4.75/7) and familiar (4.83/7), the rotation technique *Dual-Finger Rotation* was easy to learn (5.17/7) and familiar (5.42/7), and the navigation technique *Dual-Finger Navigation* was easy to learn (5.5/7) and familiar (5/7).

Participants usually felt stressed while selecting small targets using *Tapping*, whereas few participants told that *Dual-Finger Midpoint Ray-Casting* was much easier to use. With the midpoint technique, it was hard for the participants to choose objects near the edges of the display but several participants reported that it was easier using the offset technique. The participants reported that the *Dual-Finger Midpoint Ray-Casting* technique felt marginally more comfortable and equally familiar to the *Tapping* technique. The results of this subjective evaluation are presented in Figs. 19, 20, 21, and 22.

Physical *Go–Go* was the least preferred technique for object manipulation. Separation of 3DOF as 2DOF on *x–z* axes and 1DOF on the *y* axis in *Dual-Finger Positioning* was more preferable than *Z-Technique*'s 3DOF separation as 2DOF on *x–y* axes and 1DOF on *z* axis. Our method of DOF separation felt easier to use and more natural to the participants. The *Dual-Finger Rotation* technique generated more interest from participants compared to the widespread *Arc-Ball* (Figs. 1, 2).

While navigating in the environment, our *Dual-Finger Navigation* technique felt the easiest to use but the hybrid *Pointing* technique felt more familiar to the participants with a marginal difference, due to its physical viewpoint rotation.

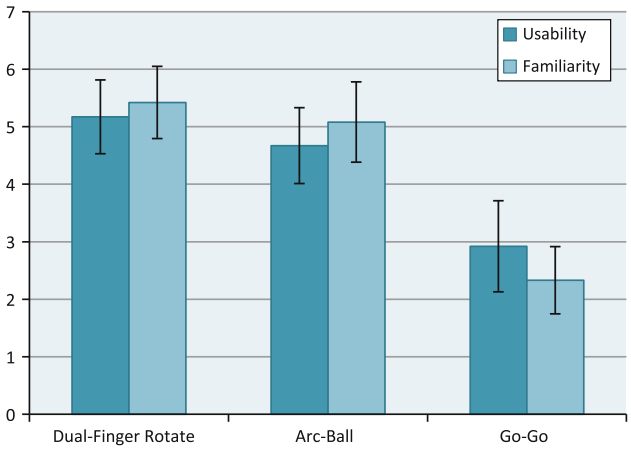

Fig. 21 Subjective evaluation of object rotation techniques. *Error bars* represent 95 % confidence interval

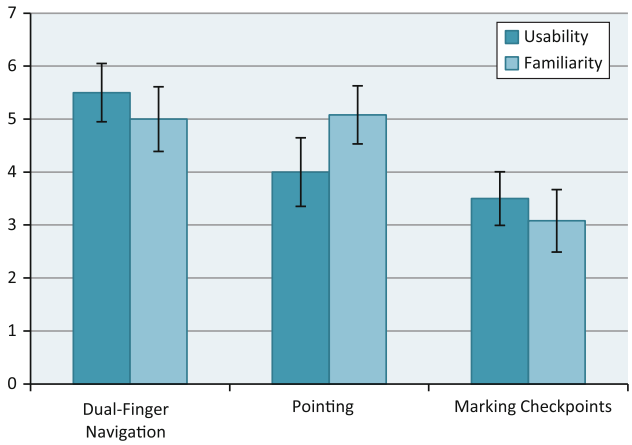

Fig. 22 Subjective evaluation of navigation techniques. *Error bars* represent 95 % confidence interval

The evaluation provides us results with which to compare the old idea for the familiarity of an interaction technique, related to the use of strong metaphors, and the new idea of naturalness and ease of use.

8 Conclusions

We have presented a set of 3D interaction techniques for mobile devices, including two high-speed and precise selection techniques: *Dual-Finger Midpoint Ray-Casting* and *Dual-Finger Offset Ray-Casting*. Our methods are able to yield fast and accurate results for the three object selection complexities that users are likely to encounter in any virtual environment. We also present an accurate and quick object positioning technique (*Dual-Finger Translation*), which decomposes a 3DOF positioning task into a set of 2DOF and 1DOF precise positioning tasks, and an accurate and quick object rotating technique (*Dual-Finger Rotate*) that separates the 3DOF task into three 1DOF subtasks to avoid being error-prone. Finally, we present a navigation technique (*Dual-Finger Navigation*) that helps users easily perform movement and viewpoint direction changes on a touch-screen without releasing their fingers.

The controlled experiment results show that dual-finger interaction provides a feasible solution for increasing the precision and speed of universal 3D interaction tasks—object selection, manipulation, and navigation—on handheld devices. The limitations of the mobile devices, including the small physical screen size and limited input modalities, combined with higher complexity of the virtual environment, such as highly occluded or small-sized objects, could be overcome to a great extent with dual-finger interaction. The subjective evaluation results also reveal that this type of interaction has the potential to increase the overall usability of 3D applications. Rather than gestural interaction or on-screen simulation of game pads, more commonly preferred solutions in today's touch-based 3D applications, our user study shows that 3D interaction can be done directly, considering the ease of use and universality of the solutions.

Our experimental study also reveals that existing immersive and desktop 3D virtual environment techniques, such as *Ray-Casting*, *Go–Go*, *Occlusion* techniques, perform less effectively on handheld devices. This disadvantage is more prominent while selecting and manipulating smaller objects, or interacting in complex virtual environments. Furthermore, the experimental results show that the current rule of thumb ("perform small tasks physically and bigger tasks virtually") for 3D interaction is not appropriate for interacting in virtual environments on mobile devices. We also find that decomposing 3DOF into smaller DOFs does not result in task completion latency and it yields lower error rates, validating this finding also in the mobile context.

Our investigation of existing techniques suggests that there is room for further research; thus, new 3D mobile interaction techniques are likely to emerge in the near future. Particularly, precise selection techniques on mobile displays are an important issue that needs to be dealt with efficiently. We offer two new selection techniques for this purpose; however, for the sake of usability, we have strayed from the principle of directness and direct manipulation of an object by directly touching it. These techniques, offered for higher usability, require the use of novel metaphors and new direct manipulation techniques.

There are potential limitations of our method. Particularly, objects near the screen corners and edges are difficult to select and manipulate with two fingers. However, considering that many 3D applications assume navigation in the virtual environment, the user can easily rotate their view point toward the object of attention. An extension of our method is to automatically rotate the view point when both fingers of the user are close to the same corner of edge. However, we have excluded this type of interaction in our user studies, to be able to better measure the independent performance of our techniques and verify our hypotheses.

Mobile devices' hardware capability has an important effect on the design of the techniques. For example, several recent hardware studies consider the finger area and not a single-touch point per finger, as an input [4, 28]. However, since currently available smartphones do not have the functionality to capture data for all touch coordinates covered by the finger, we are not able to assume such source of input. For example, it is possible to use the finger area on the touch-screen to produce an easy clicking gesture, eliminating the need for the final touch for confirmation [4]. It is possible to extend our methods to use touch pressure or area sensor input will modify our methods when available, for example, eliminating the need for the final touch for validating the action.

Acknowledgments The authors would like to thank Rana Nelson for proofreading. This work is supported by the European Commission, 3DPHONE project (FP7-213349).

References

1. Shneiderman B (2003) Why not make interfaces better than 3D reality? IEEE Comput Graph Appl 23:12–15
2. Bowman DA, Kruijff E, LaViola JJ, Poupyrev I (2004) 3D user interfaces: theory and practice. Addison Wesley Longman Publishing Co., Inc., Redwood City
3. Pierce J, Forsberg A, Conway M, Hong S, Zeleznik R, Mine M (1997) Image plane interaction techniques in 3D immersive environments. In: Proceedings of the 1997 symposium on Interactive 3D graphics (I3D '97), pp 39–43

4. Benko H, Wilson A, Baudisch P (2006) Precise selection techniques for multi-touch screens. In: Proceedings of the SIGCHI conference on Human Factors in computing systems (CHI '06), pp 1263–1272
5. Poupyrev I, Ichikawa T, Weghorst S, Billinghurst M (1998) Egocentric object manipulation in virtual environments: empirical evaluation of interaction techniques. Comput Graph Forum 17(3):41–52
6. Stoakley R, Conway M, Pausch R (1995) Virtual reality on a WIM: interactive worlds in miniature. In: Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '95), pp 265–272
7. Forsberg A, Herndon K, Zeleznik R (1996) Aperture based selection for immersive virtual environments. In: Proceedings of the 9th annual ACM symposium on User interface software and technology (UIST '96), pp 95–96
8. Bowman D, Hodges L (1997) An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In: Proceedings of the 1997 symposium on Interactive 3D graphics (I3D '97), pp 35–38
9. Poupyrev I, Billinghurst M, Weghorst S, Ichikawa T (1996) The go-go interaction technique: non-linear mapping for direct manipulation in VR. In: Proceedings of the 9th annual ACM symposium on User interface software and technology (UIST '96), pp 79–80
10. Bowman D, Koller D, Hodges L (1998) A Methodology for the evaluation of travel techniques for immersive virtual environments. Virtual Real J, Springer, pp 120–131
11. Tan D, Robertson G, Czerwinski M (2001) Exploring 3D navigation: combining speed-coupled flying with orbiting. In: Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '01), pp 418–425
12. Fu C, Goh W, Allen J (2010) Multi-touch techniques for exploring large-scale 3D astrophysical simulations. In: Proceedings of the 28th international conference on Human factors in computing systems (CHI '10), pp 2213–2222
13. Knödel S, Hachet M, Guitton P (2008) Navidget for immersive virtual environments. In: Proceedings of the 2008 ACM symposium on Virtual reality software and technology (VRST '08), pp 47–50
14. Decle F, Hachet M (2009) A study of direct versus planned 3D camera manipulation on touch-based mobile phones. In: Proceedings of the 11th international conference on human-computer interaction with mobile devices and services (MobileHCI '09), (32)
15. Wolfgang Hürst W, Bilyalov T (2010) Dynamic versus static peephole navigation of VR panoramas on handheld devices. In: Proceedings of the 9th international conference on Mobile and Ubiquitous Multimedia (MUM '10). ACM, New York, Article 25
16. Wilson A, Izadi S, Hilliges O, Garcia-Mendoza A, Kirk D (2008) Bringing physics to the surface. In: Proceedings of the 21st annual ACM symposium on User interface software and technology (UIST '08), pp 67–76
17. Hilliges O, Izadi S, Wilson A, Hodges S, Garcia-Mendoza A, Butz A (2009) Interactions in the air: adding further depth to interactive tabletops. In: Proceedings of the 22nd annual ACM symposium on User interface software and technology (UIST '09), pp 139–148
18. Agarawala A, Balakrishnan R (2006). Keepin' it real: pushing the desktop metaphor with physics, piles and the pen. In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI '06), pp 1283–1292
19. Hancock M, Carpendale S, 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 (CHI '07), pp 1147–1156
20. Reisman J, Davidson P, Han J (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 (UIST '09), pp 69–78
21. Martinet A, Casiez G, Grisoni L (2012) Integrality and separability of multitouch interaction techniques in 3D manipulation tasks. IEEE Trans Vis Comput Graph 18(3):369–380
22. Cohé A, Dècle F, Hachet M (2011) tBox: a 3d transformation widget designed for touch-screens. In: Proceedings of the 2011 annual conference on human factors in computing systems (CHI '11), pp 3005–3008
23. Henrysson A, Billinghurst M, and Ollila M (2005) Virtual object manipulation using a mobile phone. In: Proceedings of the 2005 international conference on Augmented tele-existence (ICAT '05), pp 164–171
24. Martinet A, Casiez G, Grisoni L (2009) 3D positioning techniques for multi-touch displays. In: Proceedings of the 16th ACM symposium on virtual reality software and technology (VRST '09), pp 227–228
25. Saffer Dan (2008) Designing gestural interfaces: touchscreens and interactive devices. O'Reilly Media, Inc., Sebastopol
26. Bowman D, Hodges L (1999) Formalizing the design, evaluation, and application of interaction techniques for immersive virtual environments. J Vis Lang Comput, pp 37–53
27. Herndon K, van Dam A, Gleicher M (1994) The challenges of 3D interaction: a CHI '94 workshop. SIGCHI Bull 26:36–43
28. Miyaki T, Rekimoto J (2009) GraspZoom: zooming and scrolling control model for single-handed mobile interaction. In: Proceedings of the 11th international conference on human-computer interaction with mobile devices and services (MobileHCI '09)
29. Foley J, van Dam A, Feiner S, Hughes J (1996) Computer graphics: principles and practice. Addison-Wesley Longman Publishing Co, Inc., Boston
30. Inc. Apple. (2012) iPhone 4 Tech Specs. http://www.apple.com/iphone/iphone-4/specs.html. Accessed 22 February 2012
31. Hollings B (2012) Cocos3D application development framework. The Brenwill workshop. http://brenwill.com/cocos3d/. Accessed 22 February 2012
32. Inc. Apple. (2012) Apple Xcode 3.2.6 SDK. Developer Tools. https://developer.apple.com/technologies/tools/. Accessed 22 February 2012
33. Inc. Apple (2012) iOS human interface guidelines. iOS developer library. http://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/MobileHIG/Introduction/Introduction.html Accessed 22 February 2012
34. Mendez E, Schmalstieg D (2009) Importance masks for revealing occluded objects in augmented reality. In: Proceedings of the 16th ACM Symposium on virtual reality software and technology (VRST '09), pp 247–248