Analysis of Locally Coupled 3D Manipulation Mappings Based on Mobile Device Motion

Paul Issartel, Florimond Guéniat, Tobias Isenberg, Mehdi Ammi

Analysis of Locally Coupled 3D Manipulation Mappings Based on Mobile Device Motion

Paul Issartel[1], Florimond Guéniat[2], Tobias Isenberg[3], and Mehdi Ammi[4]

[1]LIMSI-CNRS, Univ. Paris-Sud, paul.issartel@limsi.fr
[2]Dept. of Mathematics, Florida State University, contact@gueniat.fr
[3]INRIA Saclay, tobias.isenberg@inria.fr
[4]LIMSI-CNRS, Univ. Paris-Sud, mehdi.ammi@limsi.fr

Abstract—We examine a class of techniques for 3D object manipulation on mobile devices, in which the device's physical motion is applied to 3D objects displayed on the device itself. Our work focuses specifically on the *mapping* between device motion and object motion, and the specific challenges that arise from this locally-coupled configuration. We review existing manipulation techniques and introduce a formal description of the main mappings under a common notation. Based on this notation, we analyze these mappings and their properties in order to answer crucial usability questions. We first investigate how the 3D objects should move on the screen, since the screen also moves with the mobile device during manipulation. We then investigate the effects of a limited range of manipulation and present a number of solutions to overcome this constraint. This work provides a theoretical framework to better understand the properties of locally-coupled 3D manipulation mappings based on mobile device motion.

1 Introduction

Mobile devices differ from traditional computers in that they combine input, display, and processing capabilities into a single handheld object. Recent technological advances have made it possible to run 3D applications directly on mobile devices. One of the fundamental tasks (Bowman et al., 2004) in such applications is object manipulation, i. e. the translation and rotation of objects in 3D space. A major challenge for 3D manipulation tool design is thus to create efficient 3D manipulation techniques, tailored to the unique characteristics of this portable and integrated environment.

Currently, the most common way to interact with mobile devices is by means of an integrated touch screen. Each contact on a touch screen provides two degrees of freedom (DOF). While this type of input is well suited to 2D interaction, 3D manipulation requires three degrees of freedom for translations and three for rotations. The constraint of 2DOF input often leads to complex and unnatural 3D manipulation techniques. An alternative type of input exists in the form of tangible interaction: manipulating physical objects around the mobile device (Issartel et al., 2014). The motion of these physical objects is then mapped to the 3D objects displayed on the device's screen. Tangible input integrates all six degrees of freedom required for 3D interaction into a simple and natural way that takes advantage of real-world manipulation skills (Ishii, 2008). One important drawback, though, is that the user must carry and handle several objects in addition to the mobile device.

In this paper, we investigate a different class of techniques which retain the advantages of tangible interaction but do not require any external objects. They consist in using the *mobile device itself* as a tangible input device, by measuring its own motion relative to the environment. In other words, these techniques use the physical motion of the mobile device in the real world to control a 3D object on the device's screen. Compared to the previously mentioned interaction modes, this approach has clear advantages. Unlike touch input, it provides sufficient degrees of freedom for 3D interaction. Unlike the tangible interfaces described above, it does not require any separate objects.

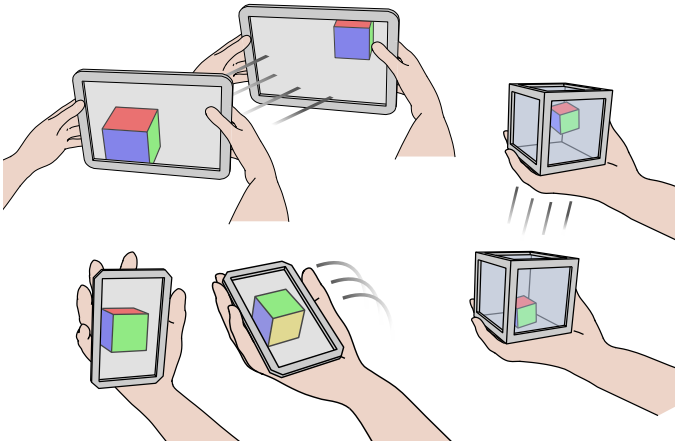

Figure 1: Using the motion of a mobile device to translate and rotate a 3D object displayed on the device itself (locally coupled manipulation). This is illustrated here on different types of mobile devices. In this paper, we specifically focus on the *mapping* between device motion and object motion.

However, this configuration also presents important challenges. The screen on which the manipulated object is displayed is coupled with the input device—a "locally coupled" configuration (Rahman et al., 2009). Therefore, *the screen moves and rotates along with the device*. This raises crucial usability questions. The first question is how to match visual feedback to device motion. Objects displayed on the screen appear to move from the user's point of view, since the screen is moving during manipulation. This raises the issue of how the manipulated object should move on the *screen* itself, so that its apparent motion remains consistent with the device's motion. Another question is whether users see the device as a "handle" that controls the object, or a "window" that controls the

viewpoint. A third issue is the limited range of manipulation. As with any handheld input device, this range is limited by the space reachable by the user. But the screen is attached to the device itself. Since the screen provides visual feedback, it must remain legible during manipulation, which further reduces the usable range of motion.

In order to address the previous questions, it is essential to understand well the *mapping* between device motion and object motion. In this analysis, we thus specifically focus on the mappings themselves. As we will see, several researchers have proposed manipulation techniques that were based on mobile device motion. Many of them, however, have emphasized the application rather than the mapping. We thus aim to provide an explicit discussion and detailed description of the possible mappings, facilitating a comprehensive understanding of their properties.

In this work, we contribute a *theoretical framework* for locally-coupled 3D manipulation mappings based on mobile device motion. We begin with a review of existing manipulation techniques, followed by a discussion of their common aspects. We then introduce a formalization of the main mappings and unify them under a common notation. Using this formalism, we proceed with an analysis of these mappings in order to demonstrate their properties. Our analysis addresses two main questions: how the object should move on the screen to match device motion, and how to address the constraints of a limited motion space. For each property of the mappings, we first examine existing arguments from previous work. However, where previous evidence is lacking or inconclusive, we contribute new theoretical and experimental results to answer the above questions. Based on this analysis, we finally discuss possible adaptations and improvements for each mapping. By providing a comprehensive, formalized, and substantiated overview of these mappings, our framework assists designers in making more informed choices when implementing such techniques.

2 Existing manipulation techniques

As a first step to establish our theoretical framework, we review existing 3D manipulation techniques based on mobile device motion.

2.1 3D manipulation through physical objects

The idea of using a handheld physical object—in this case, a mobile device—to manipulate virtual 3D objects can be related to graspable user interfaces (Fitzmaurice, 1996) and, more generally, to tangible interaction (Ishii, 2008). One of the earliest examples was the PassProps prototype by Hinckley et al. (1994) in which tangible objects are tracked in real space and their position and orientation are mapped to 3D objects shown on an external display. Similar examples are the Cubic Mouse (Fröhlich et al., 2000) and the CID device (van Rhijn and Mulder, 2006). The use of tangible objects for manipulation is a rather natural mode of interaction since it exploits the user's real-world manipulation skills (Ishii, 2008). The projects mentioned above, however, require custom-made objects and specific sensors for input and tracking.

With the increasing availability of mobile devices, many projects have proposed to use handhelds as readily-available tangible objects with built-in sensors (Katzakis and Hori, 2009; Ha and Woo, 2013; Benzina et al., 2012; Song et al., 2011; Ha and Woo, 2011; Liang, 2013; Du et al., 2011). These interfaces allow users to manipulate virtual 3D objects through the motion of a mobile device. Although device motion provides interactive control, the manipulated objects are still displayed on an external screen. Thus, the manipulation does not actually occur on the mobile device itself.

Alternatively, tangible objects can be used in combination with a mobile device (Issartel et al., 2014; Liang, 2013): tangible objects serve as input, while the mobile device processes and renders the manipulated 3D objects on its integrated screen. With this approach, the manipulation takes place on the mobile device as the entire interface is portable and self-contained. However, the user also has to handle several objects during the manipulation which can be ergonomically challenging. Moreover, external tangible objects need to be inconveniently carried with the mobile device to wherever the interface is used.

The next logical step is to use a mobile device as tangible input to manipulate objects displayed *on* the device. We survey these types of approaches and discuss them within our framework.

2.2 On-device interaction based on device motion

A number of existing mobile interaction techniques exploit the motion of a mobile device to translate and rotate objects on its own screen. Many such techniques are tailored for 1D or 2D interaction, but some of them are actually designed for 3D manipulation.

2.2.1 Tilt-based interaction

In one of the first works on the subject, Rekimoto (1996) proposed to use device inclination ("tilt-based interaction") to navigate menus on a palmtop computer. According to the given description, the current position within the menu directly depends on the device angle. Weberg et al. (2001) also described an interface that uses the device's tilt to navigate menus and select menu items on a PDA device. In this case, however, the device inclination controls the *rate* of motion within the menu. Oakley and O'Modhrain (2005) evaluated both approaches for menu selection. We can thus identify two ways of mapping mobile device motion to a manipulated object: one that directly controls the position of the object (position control), and another that controls its rate of motion (rate control).

Many other works have investigated tilt-based interaction. Scrolling in lists, documents, and images by tilting a mobile device seems to be a frequently studied task. Early works (Small and Ishii, 1997; Harrison et al., 1998; Bartlett, 2000) appear to use rate control, but the exact mapping is only informally described. Unfortunately, the lack of formalization makes these mappings ambiguous and difficult to compare to each other. Subsequent works on tilt-to-scroll (Hinckley et al., 2000; Eslambolchilar and Murray-Smith, 2008; Cho et al., 2007) then introduced more formally described rate control mappings. Rahman et al. (2009) present a thorough study of tilt-based position control mappings for 1-DOF discrete input. Tilt-based interaction has also been used for 2D panning and zooming. The RotoView technique (Feinstein, 2002), for example, facilitates map navigation with a rate control mapping. Joshi et al. (2012) present a hybrid position-rate control mapping to visualize 360° panoramas. Finally, tilt-based interaction has been studied for pointing. Tsandilas et al. (2013) compared rate control, position control and hybrid control for 1D pointing, with formal descriptions of each mapping. Teather and MacKenzie (2014) compared position control and rate control mappings for a 2D pointing task. This task is closer to a 3D manipulation than previous examples, since it involves accurate manipulation of an object (pointer) on the screen with multiple degrees of freedom.

2.2.2 Spatially-aware displays

The tilt-based techniques mentioned so far only use device orientation as input. Interfaces where the position of a mobile device serves as an input modality tend to be categorized as *spatially-aware displays*. For example, Small and Ishii (1997) presented a system to visualize long paintings, using a wheel-mounted monitor which scrolls its contents when rolled on the floor. Its mapping is not described in detail but appears to be position-controlled. Yee (2003) presented the "peephole display" in which movements of a PDA—tracked with tethers—allow the user to pan and navigate workspaces larger than the device's screen. Again, the mapping is position-controlled but not formally described. Spindler et al. (2014) demonstrated a similar approach with an infrared-tracked mobile device. Wang et al. (2006) used a mobile device's internal camera to track its own translations and rotations, mapping them to various 2D interaction tasks. In one of the only works to mention both position and rate control mappings in a spatially-aware display, Hansen et al. (2006) also used the integrated camera to track the device position and orientation, for several possible applications.

Overall, there seems to be fewer works that exploit device position than device orientation. This fact may be due to the complexity of tracking a position compared to an orientation. The device orientation can be easily tracked with integrated and inexpensive sensors, such as gyroscopes and magnetometers. Such sensors have long been found in many mobile devices. In contrast, tracking the device position is more difficult. Some of the above projects use wheels, wires, or external infrared (IR) sensors which are unwieldy and impractical in a mobile setting. Other projects choose to use an integrated camera. Now that cameras are becoming ubiquitous and embedded processing power becomes sufficient for real-time image analysis, inside-out optical tracking seems to be the most promising solution for small scale position tracking on a mobile device. The recently launched Tango project[1], a tactile tablet featuring inside-out motion tracking, may open the way for more applications of position tracking.

2.2.3 3D manipulation based on device motion

Although the mapping of device motion to 1D or 2D tasks can serve as a basis for 3D manipulation mappings, there is no substitute for studies focusing on actual 3D tasks. Only such studies can highlight the constraints and challenges specific to 3D interaction.

Fitzmaurice et al. (1993) described the Chameleon system in which the position of a handheld monitor controls the viewpoint on a displayed 3D scene. Subsequent works (Tsang et al., 2002) later improved this concept by tracking the device orientation in addition to its position, facilitating a full control of the viewpoint. These projects, however, primarily simulated a window on a virtual scene—restricting the possible mappings to an isomorphic position control and excluding other mappings that might be useful for 3D object manipulation.

Other projects use the motion of a mobile device for actual 3D manipulation. Some of them demonstrate 3D object manipulation in augmented reality (AR). Henrysson et al. (2005) and Marzo et al. (2014) described a "grasping" metaphor in which, during the manipulation, the object remains fixed relative to the mobile device. A drawback of this approach is that it makes it difficult to rotate the manipulated object without translating it. Since the virtual scene is fixed in an external reference frame and the manipulated object is fixed in the device reference frame, the mobile device must be moved across an arc. The HOMER-S technique (Mossel et al., 2013) eliminates this issue by separately applying device rotations to the manipulated object. As a consequence, however, the object is no longer fixed relative to the mobile device and can thus leave the field of view during large rotations. These approaches cannot avoid both of these problems as they are caused by the intrinsic separation between the object's reference frame and the device's reference frame in normal perspective rendering. A different approach is the concept proposed by Spindler et al. (2012) which uses an head-coupled perspective to let the device intersect the manipulated object, thus greatly reducing its separation from the object. Assuming head tracking is available—which can be challenging to accomplish in a truly mobile interface—this approach can solve the rotation issue. Yet, all the "grasping" techniques share another drawback: the object must remain fixed relative to the device, thus the translation mapping is restricted to isomorphic position control even though different mappings might be desirable in some situations (Section 6).

The alternative is to perform 3D manipulation entirely in the device reference frame, i. e. in screen space, avoiding the constraints caused by an external reference frame. Kratz and Rohs (2010) compared a tilt-based rotation mapping with a two-sided touch metaphor on a smartphone. Their tilt-based mapping uses rate control but only supports rotation on two axes. Neale et al. (2013) presented an interface to visualize museum artifacts on a tactile tablet. They compared touchscreen input to both tilt-based position control and rate control mappings. This interface, however, only supports object rotation and the mappings are not described in detail. Daiber et al. (2012) presented an interface to translate and rotate 3D objects on a smartphone. The tilt-based rotation mapping appears to be position-controlled. Their translations, however, are not based on device motion but on touch gestures. The PDDM device by Noma et al. (1996) is a rare example of using both device translations and rotations for screen-space 3D manipulation. The device is a palmtop monitor mounted on a mechanical arm. The authors presented four different mappings for screen-space object manipulation, all based on position control. The mappings are explained and illustrated, but not formally described. Furthermore, the study of the mappings themselves was still limited in scope. Important questions such as the frame of reference of manipulation were only mentioned as future work.

As we can see, a few existing works use the motion of a mobile device for actual screen-space 3D manipulation. But each of them only addresses a small subset of the possible mappings. Some only consider rotations and ignore translations, others only use position control, and yet others only consider rate control. The authors generally do not provide a formal description of the proposed mappings, making it difficult to generalize the results. In particular, the lack of a formal notation makes it impossible to assess key usability properties such as the matching between visual feedback and device motion and how well the proposed mappings make use of the limited motion space. In the rest of the paper we, therefore, conduct an in-depth analysis of the mappings themselves and their properties. We start by presenting a formalization of the main mappings and then use this notation to determine how well they address the above-mentioned usability questions.

3 About the term "mobile device"

Early approaches that used device motion as an input modality associated the mobile device concept with technologies such as

1. http://www.google.com/atap/project-tango/

portable TVs, PDAs, and palmtop computers. Today, the term "mobile device" generally refers to smartphones, tablets, phablets, or a size variation thereof. These devices all share a similar form factor: they are mostly flat, rectangular, and have a single screen on one side.

There is no reason, however, why a 3D manipulation mapping could not work with other device shapes (e. g., Figure 1). There have been proposals for adding a screen on the other side of current mobile devices (e. g., Kratz and Rohs, 2010), for creating highly deformable mobile devices (e. g., Ramakers et al., 2014), and for creating tiltable devices (e. g., Alexander et al., 2012). The recent interest for "smart watches" is driving the industry and academia toward the development of wristband-shaped displays (e. g., Lyons et al., 2012). There are prototypes of small, portable cubic displays (Lopez-Gulliver et al., 2009; Stavness et al., 2010) with a screen on each face, capable of displaying a 3D scene as if it were inside the cube. Spherical screens (Benko et al., 2008) and volumetric globes (Grossman and Balakrishnan, 2006) are also being investigated. These remain too large to be considered "mobile", but could be down-sized as technology advances. Future mobile devices might thus have a radically different shape than current ones.

In this article we thus define a mobile device in a rather generic way: any interactive physical (tangible) object that is capable of displaying virtual objects on its surface or inside its volume. All our conclusions remain applicable to any device which corresponds to this definition, unless otherwise specified.

There is still an additional requirement for a mobile device to be compatible with the 3D manipulation mappings discussed here. The manipulated virtual object *must* appear to have a single defined position and orientation within the device reference frame. The reason for this additional constraint is that it would be impossible to know the exact location of the object, and object manipulation would no longer make sense, if multiple copies of a single virtual object were to appear at conflicting locations.

This constraint, however, normally does not pose a problem for devices that have a single flat screen since there is only one view of the virtual scene. For devices with multiple non-coplanar screens or devices covered with a curved screen, special care must be taken to ensure that a virtual object does not appear at multiple conflicting locations. This can be accomplished with perspective correction, i. e. by ensuring that each screen (or each point of the surface) shows the virtual scene from a different perspective such that a virtual object appears at a fixed location within the device volume. This solution requires either autostereoscopic displays (Lopez-Gulliver et al., 2009), true volumetric displays (Grossman and Balakrishnan, 2006), or a way to continuously track the position of the user's eyes in order to update the perspective (Stavness et al., 2010).

4 Formalization of the Main Mappings

A *mapping*, also called transfer function, describes how device motion is mapped to object motion on the screen and we now present the main ways to perform such a mapping. We express them in an unified formal notation, allowing us to compare them and assess their properties in the next sections. Unlike many previous works, we consider both translations and rotations in our formal model. We also provide a pseudocode description in the appendix.

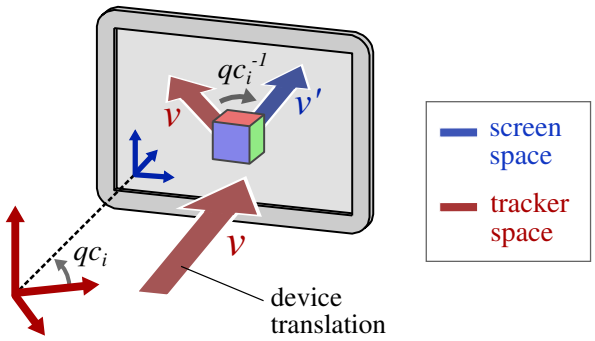

Figure 2: Directly applying the measured device translation v to the manipulated object would move it in unexpected directions, depending on the device orientation qc_i in tracker space. To get a more predictable behavior, the vector v should be rotated by the inverse rotation qc_i^{-1}, producing the correct translation v'. The same process is applied to rotations.

4.1 Basic notation

The values pc_t and qc_t represent the position and orientation, respectively, of the mobile device at time t. They are the *control* values, obtained from tracking and expressed in an arbitrary tracking-specific coordinate system. The position pc_t is a 3D vector, while the orientation qc_t is a quaternion that represents the rotation of the device relative to some base orientation.

The values pd_t and qd_t represent the position and orientation of the manipulated object at time t. They are the *display* values, expressed in the screen coordinate system. The position pd_t is a 3D vector, while the orientation qd_t is a quaternion that represents the rotation of the object relative to some base orientation on the screen. The display values pd_t and qd_t are computed from the control values by applying the mapping function.

Time $t=0$ designates the beginning of manipulation, i. e. the time when the user starts manipulating the object.[2] The values pc_0 and qc_0 thus represent the initial position and orientation of the mobile device. Similarly, pd_0 and qd_0 represent the initial position and orientation of the manipulated object. Each subsequent time t indicates the time when a new sample is obtained from the tracking system. Time increments are unitary in our notation.

4.2 From tracker coordinates to screen coordinates

Control values (positions pc_t and orientations qc_t of the mobile device) are measured by the tracking system in a tracking-specific reference frame. Consequently, the motion of the mobile device is also expressed in this reference frame. But the manipulated object belongs to the screen reference frame. Therefore, the mapping function must convert device motion into the screen reference frame.

In the locally coupled configuration we study here, the screen is attached to the input device itself. Thus, a rotation of the device during manipulation also rotates the screen in relation to the tracking system, causing the screen and tracking reference frames to become *misaligned* (Figure 2). If the measured device motion was directly applied to an object on the screen, the object would move in unexpected directions. Converting device motion into the screen reference frame requires to compensate for this misalignment.

2. Users should be able to explicitly engage or disengage manipulation mode with, e. g., a dedicated button to move the device without affecting the object.

Consider a translation v and a rotation r of the mobile device, measured in the tracking reference frame between times i and j. At the beginning of movement, the device orientation in the tracking reference frame is qc_i. Since the screen is attached to the mobile device, the screen orientation is also qc_i (for the sake of simplicity, we assume a null offset between the device and the screen). To map v and r to the screen reference frame, this orientation must be *canceled*, hence re-aligning the reference frames (Figure 2). We achieve this re-alignment by applying the inverse rotation qc_i^{-1} to v and r. We thus apply this inverse rotation to the direction (the vector part) of the quaternion r using the conjugation operation $qc_i^{-1}\ r\ (qc_i^{-1})^{-1}$, shortened to $qc_i^{-1}\ r\ qc_i$. For this purpose we consider the translation vector v as a quaternion whose real part is zero, and apply the same operation. In summary, the new translation v' and rotation r' (corresponding to the translation v and the rotation r measured by the tracking system) are hence obtained as follows:

$$\begin{aligned} v' &= qc_i^{-1}\ v\ qc_i \\ r' &= qc_i^{-1}\ r\ qc_i \end{aligned} \tag{1}$$

This transformation expresses device translations and rotations in a stable reference frame. We can now apply these transformations to a 3D object displayed on the screen. As demonstrated in previous work, however, there are different ways to apply device motion to a manipulated object (Figure 3) as we will see next.

4.3 Position control mappings

In a "position control" (or zero-order) mapping, the motion of an input device directly controls the position and orientation of the manipulated object (Zhai, 1995). In our case this means that translations of the mobile device control the position of the displayed object, while rotations of the device control the orientation of the object.

Two main ways of mapping exist to control an object's position and orientation. The first one—an "absolute" mapping—directly assigns the position and orientation of the mobile device to the object, as measured in some fixed reference frame. The other way—a "relative" mapping—applies *incremental* translations and rotations of the device (i. e., its change in position and orientation between each times t and $t-1$) to the object. Our notion of absolute and relative mappings reuses the terminology proposed by Poupyrev et al. (2000) for rotations, which we extend to also include translations.

Absolute and relative sensors The distinction between absolute and relative mappings has a practical significance. Some tracking sensors measure an "absolute" position or orientation, i. e. expressed in a static reference frame outside of the mobile device. For example, a mechanical arm can measure the location of the device relative to its base, and a magnetometer can measure the orientation of the device in the Earth's reference frame. An embedded camera can track a fixed marker in the environment (Hansen et al., 2006) or sense IR light reflected from the surrounding environment (Tango project) to measure the device's position and orientation. Other sensors only measure relative motion such as the gyroscopes or accelerometers found on many current devices. An absolute sensor can be used with both absolute and relative mappings, whereas a relative sensor is not suitable for absolute mappings due to drift. However, relative inside-out sensors are generally fully contained in the mobile device itself and do not depend on the external environment, which is a strong benefit for portability. Although some absolute sensors can be physically embedded in the device (e. g., cameras and magnetometers), they are easily disrupted by some environmental conditions (lack of visual markers, insufficient or excessive ambient light, presence of magnetic materials, etc.). Thus, the use of relative sensors rather than absolute ones might be dictated by technical constraints.

Aside from these practical aspects, the distinction between absolute and relative tracking also has an impact on usability since—as we demonstrate next—absolute and relative position control mappings do not have the same properties.

4.3.1 Absolute mapping

An absolute mapping (Poupyrev et al., 2000; Bowman et al., 2004) directly applies the device position and orientation (pc_t and qc_t) to the manipulated object position and orientation (pd_t and qd_t). To make incremental manipulation possible, the mapping must take into account the initial position and orientation of the object (pd_0 and qd_0). To ensure that the object does not move unexpectedly at the beginning of manipulation, it is also necessary to subtract the initial position and orientation of the mobile device (pc_0 and qc_0). This corresponds to a translation $pc_t - pc_0$ and a rotation $qc_t\ qc_0^{-1}$ which are to be applied to the object's initial position pd_0 and orientation qd_0. As we explained in Section 4.2, the device translations and rotations must be converted into screen space. Since they are measured from the initial device location at $t = 0$, a rotation of qc_0^{-1} is applied. The absolute mapping is thus given by:

$$\begin{aligned} \Delta pc_t &= qc_0^{-1}\ (pc_t - pc_0)\ qc_0 \\ \Delta qc_t &= qc_0^{-1}\ (qc_t\ qc_0^{-1})\ qc_0 \\ pd_t &= \Delta pc_t + pd_0 \\ qd_t &= \Delta qc_t\ qd_0 \end{aligned} \tag{2}$$

4.3.2 Relative mapping

Rather than directly applying the device position and orientation to the object, we can also apply *incremental* translation and rotation offsets. A relative mapping (Poupyrev et al., 2000; Bowman et al., 2004) applies incremental device translations and rotations, measured between times $t-1$ and t ($pc_t - pc_{t-1}$ and $qc_t\ qc_{t-1}^{-1}$, respectively), to the current object position and orientation (pd_{t-1} and qd_{t-1}, resp.). Again, device translations and rotations must be converted into screen space. Since they are measured from time $t-1$, a rotation of qc_{t-1}^{-1} is applied. The relative mapping is thus given by:

$$\begin{aligned} \Delta pc_t &= qc_{t-1}^{-1}\ (pc_t - pc_{t-1})\ qc_{t-1} \\ \Delta qc_t &= qc_{t-1}^{-1}\ (qc_t\ qc_{t-1}^{-1})\ qc_{t-1} \\ pd_t &= \Delta pc_t + pd_{t-1} \\ qd_t &= \Delta qc_t\ qd_{t-1} \end{aligned} \tag{3}$$

In order to unify all the main mappings under a common formalism, our notation assumes the availability of *absolute* tracking information. However, relative sensors embedded into the device (such as gyroscopes) provide *incremental* device translations and rotations. Since those incremental translations and rotations are already expressed in the device's reference frame, they do not have to be converted into screen space. Therefore, the values returned by such sensors can be directly used in place of the Δpc_t and Δqc_t terms in Eq. 3.

As a consequence, only the relative position-control mapping should be used with relative sensors, since it does not require the

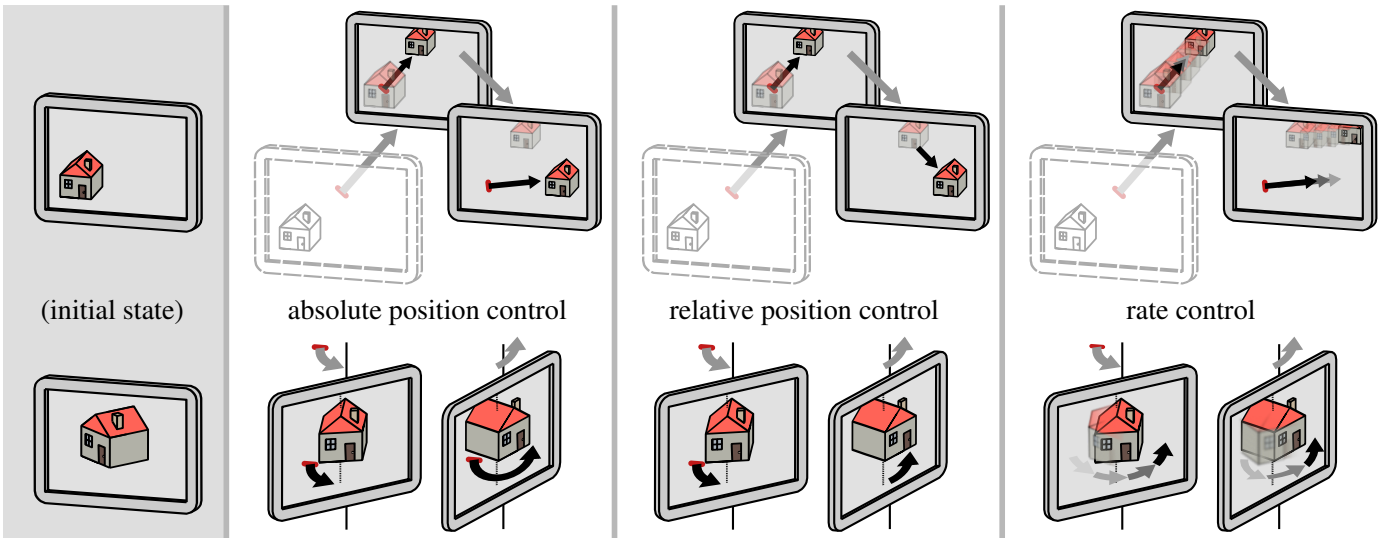

Figure 3: Three main ways to map mobile device motion to a 3D object's motion, shown for a tablet. Absolute position control: the device displacement *from its initial location* is applied to the object. Relative position control: the *incremental* device displacement is applied to the object. Rate control: the device displacement from its initial location controls the object's *velocity*.

absolute pc_t and qc_t values to be known (unless the sensor values are integrated, ultimately leading to drift).

4.4 Rate control mapping

In a "rate control" (or first-order) mapping, the motion of the input device controls the *velocity* (linear or angular) of the object (Zhai, 1995). The mobile device can be translated and rotated in 3D space from its initial position pc_0 and orientation qc_0. In a rate control mapping, the linear velocity of the manipulated object increases when the mobile device moves away from its initial position, and decreases when returning to this point. The linear velocity thus depends on the translation vector $pc_t - pc_0$. Similarly, the angular velocity of the manipulated object depends on the rotation of the device from its initial orientation, i.e., $qc_t\, qc_0^{-1}$. Since those displacements are relative to the initial device location (at $t=0$), a rotation of qc_0^{-1} is applied to convert them to screen space. Applying a linear and angular velocity to an object means adding these to its current position and orientation (pd_{t-1} and qd_{t-1}) at each time t. The rate control mapping is thus given by:

$$\begin{aligned} \Delta pc_t &= qc_0^{-1}\,(pc_t - pc_0)\,qc_0 \\ \Delta qc_t &= qc_0^{-1}\,(qc_t\,qc_0^{-1})\,qc_0 \\ pd_t &= \Delta pc_t + pd_{t-1} \\ qd_t &= \Delta qc_t\,qd_{t-1} \end{aligned} \tag{4}$$

4.5 Higher-order control

Position control is a zero-order mapping: it directly maps device positions and orientations to the object. Rate control is a first-order mapping: it maps the device location to the object *velocity*, i. e. the derivative of position and orientation. While higher-order mappings such as acceleration control[3] are possible, they are known to perform worse than position control and rate control (Massimino et al., 1989; Zhai, 1995). We thus do not consider them here.

3. The metaphor for a second-order mapping would be an impulse being applied to an object that causes it to continue moving until a reverse impulse is applied—similar to what happens on space vehicles (Massimino et al., 1989).

4.6 Control-display gain

The mappings as they are formulated above do not change the scale of device movements which are applied to the object. Such mappings are called *isomorphic* (Poupyrev et al., 2000; Zhai, 1995). However, we can easily extend them to amplify or reduce translations and rotations. The resulting mappings then become *non-isomorphic*.

We thus introduce a gain function to our framework that computes a scalar *gain factor* k_t at each time t:

$$k_t = \text{gain}(t)$$

This gain factor allows us to rescale the device translations and rotations before applying them to the object, so that the object on the screen can move faster or slower than the device itself. For a translation, the gain factor changes its length without altering its direction. This is accomplished by scaling the translation vector Δpc_t by the gain factor k_t, yielding a new translation vector $\Delta pc'_t$:

$$\Delta pc'_t = k_t\,\Delta pc_t$$

In the case of a rotation of angle θ around a given axis, the gain factor changes the angle without altering the axis. If the rotation Δqc_t is expressed as a quaternion, we can use a slerp interpolation (Shoemake, 1985) from the identity quaternion $\mathbf{1}$ to construct a new rotation $\Delta qc'_t$ around the same axis but with an angle scaled by k_t. We note this operation as $\Delta qc_t^{k_t}$. If Δqc_t is a non-null rotation, the new rotation $\Delta qc'_t = \Delta qc_t^{k_t}$ is given by:

$$\begin{aligned} \Delta qc'_t &= \text{slerp}(1, \Delta qc_t, k_t) \\ &= \Delta qc_t^{k_t} \end{aligned}$$

By substituting Δpc_t and Δqc_t with $k_t\,\Delta pc_t$ and $\Delta qc_t^{k_t}$ in the mappings presented above, it becomes possible to dynamically control the gain applied to translations and rotations.

The gain factor in our model is a function of the current time. Unlike some previous works (e. g., (Teather and MacKenzie, 2014; Poupyrev et al., 2000; LaViola and Katzourin, 2007)) which only

used static scaling coefficients, we emphasize that the gain may *dynamically change* during manipulation. Such a variable gain factor is especially useful to increase the range and accuracy of manipulation as we show in Section 6.2. We thus indicate below whether the properties of each mapping remain true even with a variable gain factor.

5 SPATIAL FEEDBACK COMPLIANCE

When users are moving a mobile device to control a 3D object on the screen, they receive multiple forms of feedback. The first is kinesthetic/proprioceptive feedback from translating and rotating the device itself. The second is visual feedback from the resulting object motion on the screen. To maintain user performance and comfort it is thus essential that the visual feedback matches the kinesthetic/proprioceptive feedback (Smith and Smith, 1987)—a principle known as feedback compliance (Bowman et al., 2004). Here, we focus specifically on *spatial feedback compliance*, which refers to the motion of the virtual object and is thus especially relevant when designing mappings.

In this section we discuss the spatial compliance properties of each mapping, both for translations and for rotations. We begin with the two properties mentioned by Bowman et al. (2004), directional and nulling compliance, along with the property of transitivity (Bade et al., 2005). Finally, we address the question of the user's reference frame (allocentric or egocentric) and whether object motion matches the reference frame expected by the user.

5.1 Directional compliance

Directional compliance (Bowman et al., 2004; Poupyrev et al., 2000), also called "kinesthetic correspondence" (Britton et al., 1978) or "stimulus-response compatibility" (Fitts and Seeger, 1953), means that the manipulated object moves *along the same direction* as the controlling device. In the configuration studied here, the object moves on the screen and is controlled by the mobile device's motion. The screen itself, however, is attached to the device and is also moving during manipulation. It is thus important to consider device motion relative to the screen (i. e., in screen space). Directional compliance, therefore, means that the object is moving *on the screen* along the same direction as the device is moving *relative to the screen*.

Note that the conversion to screen space described in Section 4.2 ensures that device motion is consistently aligned with screen space at $t=0$, but does not guarantee directional compliance at any subsequent time during manipulation.

Object motion corresponds to the change of position and orientation on the screen between times $t-1$ and t: $pd_t - pd_{t-1}$ and $qd_t\, qd_{t-1}^{-1}$, resp. Mobile device motion corresponds to the change of position and orientation in tracking space between times $t-1$ and t: $pc_t - pc_{t-1}$ and $qc_t\, qc_{t-1}^{-1}$, resp. As before, a rotation of qc_{t-1}^{-1} must be applied to this device motion to convert it to screen space. Formally stated, directional compliance means that object translations are collinear with device translations relative to the screen, and that object rotations have the same axis as device rotations relative to the screen. Thus, a mapping is directionally compliant at time t if it can be expressed as:

$$\begin{aligned} &\exists\,(\alpha,\beta)\in\mathbb{R}^2: \\ &\quad pd_t - pd_{t-1} = \alpha\ \left(qc_{t-1}^{-1}\,(pc_t - pc_{t-1})\,qc_{t-1}\right) \\ &\quad qd_t\,qd_{t-1}^{-1} = \left(qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})\,qc_{t-1}\right)^{\beta} \end{aligned} \tag{5}$$

Relative position control. The relative position control mapping is always directionally compliant, for both translations and rotations. From the mapping formulation (Eq. 3), and by taking into account the gain factor (Section 4.6), we get $pd_t - pd_{t-1} = k_t\,\Delta pc_t = k_t\,\left(qc_{t-1}^{-1}\,(pc_t - pc_{t-1})\,qc_{t-1}\right)$ and $qd_t\,qd_{t-1}^{-1} = \Delta qc_t^{k_t} = \left(qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})\,qc_{t-1}\right)^{k_t}$—equivalent to the expression in Eq. 5. Relative position control thus always guarantees directional compliance for both translations and rotations.

Rate control. The rate control mapping also ensures directional compliance for translations and rotations. This mapping (Eq. 4) is equivalent to continually applying the relative mapping's first step (Eq. 3) to the object, during which the device moves directly from (pc_0, qc_0) to $(pc_1, qc_1) = (pc_t, qc_t)$. Since the relative mapping always guarantees directional compliance (including on its first step) the rate control mapping is also directionally compliant.

Absolute position control. Absolute position control mappings do *not* guarantee directional compliance *in the general case*. However, directional compliance can still be obtained under specific conditions. By taking into account a variable gain factor, we can express object translations between two times $t-1$ and t as:

$$\begin{aligned} &pd_t - pd_{t-1} \\ &= (k_t\,\Delta pc_t + pd_0) - (k_{t-1}\,\Delta pc_{t-1} + pd_0) \\ &= k_t\,\Delta pc_t - k_{t-1}\,\Delta pc_{t-1} \\ &= k_t\,\left(qc_0^{-1}\,(pc_t - pc_0)\,qc_0\right) - k_{t-1}\,\left(qc_0^{-1}\,(pc_{t-1} - pc_0)\,qc_0\right) \\ &= \left(qc_0^{-1}\,k_t\,(pc_t - pc_0)\,qc_0\right) - \left(qc_0^{-1}\,k_{t-1}\,(pc_{t-1} - pc_0)\,qc_0\right) \\ &= qc_0^{-1}\,\left(k_t\,(pc_t - pc_0) - k_{t-1}\,(pc_{t-1} - pc_0)\right)\,qc_0 \end{aligned} \tag{6}$$

Thus, in the general case, object translations do not correspond to expression Eq. 5 and are not directionally compliant. For constant gain factors k_t ($k_t = \alpha\ \forall t>0$), however, Eq. 6 can be reduced to:

$$\begin{aligned} &= qc_0^{-1}\,\left(\alpha\,(pc_t - pc_0) - \alpha\,(pc_{t-1} - pc_0)\right)\,qc_0 \\ &= qc_0^{-1}\,\alpha\,\left((pc_t - pc_0) - (pc_{t-1} - pc_0)\right)\,qc_0 \\ &= qc_0^{-1}\,\alpha\,(pc_t - pc_{t-1})\,qc_0 \\ &= \alpha\,\left(qc_0^{-1}\,(pc_t - pc_{t-1})\,qc_0\right) \end{aligned}$$

Moreover, if at $t-1$ the mobile device orientation qc_{t-1} is equal to its initial orientation qc_0 then object translations can be written as:

$$= \alpha\,\left(qc_{t-1}^{-1}\,(pc_t - pc_{t-1})\,qc_{t-1}\right)$$

This corresponds to Eq. 5. Translations in the absolute mapping are thus only directionally compliant if the gain factor remained constant between $t=0$ and $t-1$ *and* if the mobile device orientation is the same as its initial orientation. Concerning rotations, incremental object motion can be written as:

$$\begin{aligned} &qd_t\,qd_{t-1}^{-1} \\ &= (\Delta qc_t^{k_t}\,qd_0)\,(\Delta qc_{t-1}^{k_{t-1}}\,qd_0)^{-1} \\ &= (\Delta qc_t^{k_t}\,qd_0)\,(qd_0^{-1}\,\Delta qc_{t-1}^{-k_{t-1}}) \\ &= \Delta qc_t^{k_t}\,\Delta qc_{t-1}^{-k_{t-1}} \\ &= \left(qc_0^{-1}\,(qc_t\,qc_0^{-1})\,qc_0\right)^{k_t}\,\left(qc_0^{-1}\,(qc_{t-1}\,qc_0^{-1})\,qc_0\right)^{-k_{t-1}} \end{aligned} \tag{7}$$

In the general case, rotations do not correspond to the form stated in Eq. 5 and are not directionally compliant. However, if $qc_t\,qc_0^{-1}$ (device rotation from its initial orientation at time t) and $qc_{t-1}\,qc_0^{-1}$ (same at time $t-1$) have the same rotation axis then their difference $qc_t\,qc_{t-1}^{-1}$ also has the same rotation axis. There exist, therefore,

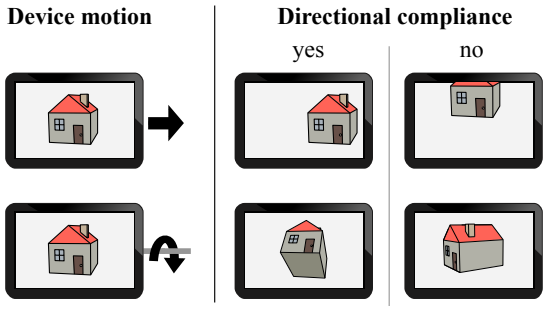

Figure 4: Directional compliance versus non-compliance, shown here on a tablet-shaped device.

two gain factors a_1 and a_2 such as $qc_t\,qc_0^{-1} = (qc_t\,qc_{t-1}^{-1})^{a_1}$ and $qc_{t-1}\,qc_0^{-1} = (qc_t\,qc_{t-1}^{-1})^{a_2}$ and Eq. 7 can be rewritten as:

$$\begin{aligned}
&= \left(qc_0^{-1}\,(qc_t\,qc_{t-1}^{-1})^{a_1}\,qc_0\right)^{k_t}\left(qc_0^{-1}\,(qc_t\,qc_{t-1}^{-1})^{a_2}\,qc_0\right)^{-k_{t-1}}\\
&= \left(qc_0^{-1}\,(qc_t\,qc_{t-1}^{-1})\,qc_0\right)^{a_1+k_t}\left(qc_0^{-1}\,(qc_t\,qc_{t-1}^{-1})\,qc_0\right)^{a_2-k_{t-1}}\\
&= \left(qc_0^{-1}\,(qc_t\,qc_{t-1}^{-1})\,qc_0\right)^{a_1+k_t+a_2-k_{t-1}}
\end{aligned}$$

We have $qc_0^{-1} = qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})^{a_2}$ due to $qc_{t-1}\,qc_0^{-1} = (qc_t\,qc_{t-1}^{-1})^{a_2}$. With $a_1 + k_t + a_2 - k_{t-1} \equiv \beta$, we can thus reduce object rotations to:

$$\begin{aligned}
&= \left(qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})^{a_2}\,(qc_t\,qc_{t-1}^{-1})\left(qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})^{a_2}\right)^{-1}\right)^{\beta}\\
&= \left(qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})^{a_2}\,(qc_t\,qc_{t-1}^{-1})\,(qc_t\,qc_{t-1}^{-1})^{-a_2}\,qc_{t-1}\right)^{\beta}\\
&= \left(qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})^{a_2+1-a_2}\,qc_{t-1}\right)^{\beta}\\
&= \left(qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})\,qc_{t-1}\right)^{\beta}
\end{aligned}$$

This corresponds to Eq. 5. Hence, object rotations in the absolute mapping are only directionally compliant along the initial rotation axis. In practice, this means that only the first rotation is directionally compliant: subsequent rotations are not if they happen on a different axis. Therefore, users who wish to rotate the object around another axis will have to return the device to its initial orientation in order to maintain directional compliance for further rotations.

5.2 Transitivity and nulling compliance

Transitivity (Bade et al., 2005) refers to a property of the real world: moving an object from point A to point B then C, or directly from A to C, results in the same final location for the object. According to this principle, translating and rotating the mobile device from A to B then C, or directly from A to C, should bring the manipulated object to the same position and orientation. In particular, this property allows users to easily return the object to its initial location—which can be useful after a manipulation error—by simply returning the mobile device to its initial location. This specific case is known as *nulling compliance* (Poupyrev et al., 2000). Transitivity is a generalization of nulling compliance to any target location.

Absolute position control. The absolute position control mapping is *generally* transitive, for both translations and rotations. The only terms in the mapping formulation (Eq. 2) that are non-constant during manipulation are pc_t and qc_t. The base formulation of the absolute mapping only depends on the *current* position and orientation of the mobile device, regardless of the intermediate steps that led it there. An isomorphic absolute mapping is thus always transitive. For a non-isomorphic mapping, however, we must also take the gain function into account. A non-isomorphic absolute mapping is only transitive if the gain function itself does not depend on non-constant terms other than pc_t and qc_t. This is obviously the case for any constant gain factor. But adding non-constant parameters to the gain function other than pc_t and qc_t breaks this guarantee. In any case, both isomorphic and non-isomorphic absolute mappings are always transitive in relation to the initial position and orientation pc_0 and qc_0 since a null translation or rotation is unaffected by gain. They are thus *always* nulling compliant. This property of absolute position control was identified by Poupyrev et al. (2000) for rotations, but it also holds for translations.

Rate control. The rate control mapping is *not* transitive or nulling compliant, neither for translations nor rotation, because it is time-dependent by definition. The resulting object motion depends on the time spent by the mobile device between locations A, B, and C. There is thus no way to predict the final location of the manipulated object from a sequence of device locations alone.

Relative position control. The relative position control mapping is *not* transitive or nulling compliant in the general case. But there are still specific conditions for which this mapping can be transitive. In order to demonstrate this, let pd_t^{step} be the position of the manipulated object obtained after the device went through intermediate positions between pc_0 and pc_t. Let pd_t^{direct} be the object position obtained after the device moved *directly* from pc_0 to $pc_1 = pc_t$. Similarly, qd_t^{step} is the object orientation obtained after the device rotated incrementally from qc_0 to qc_t, and qd_t^{direct} is the object orientation obtained after the device directly rotated from qc_0 to $qc_1 = qc_t$. Concerning the object positions, we have:

$$\begin{aligned}
pd_t^{\text{step}} &= k_t\,\left(qc_{t-1}^{-1}\,(pc_t - pc_{t-1})\,qc_{t-1}\right) + pd_{t-1}\\
&= \left(qc_{t-1}^{-1}\,k_t\,(pc_t - pc_{t-1})\,qc_{t-1}\right) + pd_{t-1}\\
&= \left(qc_{t-1}^{-1}\,k_t\,(pc_t - pc_{t-1})\,qc_{t-1}\right)\\
&\quad + \ldots + \left(qc_0^{-1}\,k_1\,(pc_1 - pc_0)\,qc_0\right) + pd_0\\
pd_t^{\text{direct}} &= k_t\,\left(qc_0^{-1}\,(pc_t - pc_0)\,qc_0\right) + pd_0\\
&= qc_0^{-1}\,k_t\,(pc_t - pc_0)\,qc_0 + pd_0
\end{aligned}$$

Due to the transformations (rotation to screen space and gain factor) applied to incremental $pc_i - pc_{i-1}$ vectors in pd_t^{step}, the result is generally not equivalent to pd_t^{direct}. This observation shows that translations are not transitive under this mapping in the general case. However, if mobile device orientation did not change so far ($qc_t = qc_0\ \ \forall t{>}0$) then pd_t^{step} is reduced to:

$$\begin{aligned}
pd_t^{\text{step}} = qc_0^{-1}\,\big(&k_t\,(pc_t - pc_{t-1})\\
&+ \ldots + k_1\,(pc_1 - pc_0)\big)\,qc_0 + pd_0
\end{aligned} \tag{8}$$

Eq. 8 applies arbitrary gain factors to each intermediate translation step. For Eq. 8 to become equivalent to pd_t^{direct}, the gain factor must also have been constant during manipulation ($k_i = k_t\ \ \forall i{<}t$):

$$\begin{aligned}
pd_t^{\text{step}} &= qc_0^{-1}\,k_t\,(pc_t - pc_0)\,qc_0 + pd_0\\
&= pd_t^{\text{direct}}
\end{aligned}$$

Therefore, translations in the relative mapping are only transitive as long as the gain factor *and* the mobile device orientation remained

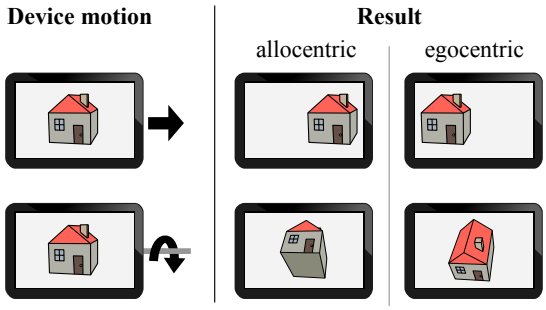

Figure 5: Difference between allocentric and egocentric manipulation, shown here on a tablet-shaped device.

constant so far during manipulation. Concerning rotations we have:

$$\begin{aligned} qd_t^{\text{step}} &= \left(qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})\,qc_{t-1}\right)^{k_t}\,qd_{t-1} \\ &= \left((qc_{t-1}^{-1}\,qc_t)\,(qc_{t-1}^{-1}\,qc_{t-1})\right)^{k_t}\,qd_{t-1} \\ &= (qc_{t-1}^{-1}\,qc_t)^{k_t}\,qd_{t-1} \\ &= (qc_{t-1}^{-1}\,qc_t)^{k_t}\,(qc_{t-2}^{-1}\,qc_{t-1})^{k_{t-1}}\ldots(qc_0^{-1}\,qc_1)^{k_1}\,qd_0 \\ qd_t^{\text{direct}} &= \left(qc_0^{-1}\,(qc_t\,qc_0^{-1})\,qc_0\right)^{k_t}\,qd_0 \\ &= (qc_0^{-1}\,qc_t)^{k_t}\,qd_0 \end{aligned}$$

Even if the gain factor remains constant, qd_t^{step} is still not equivalent to qd_t^{direct} in the general case. Rotations are thus not transitive for relative mappings (except when $k_t=-1$; see below).

5.3 Allocentric and egocentric manipulation

Translating and rotating a 3D object can be interpreted in two ways (Klatzky, 1998). One interpretation is that the object itself is moved relative to a stationary viewpoint. This type of transformation is called *allocentric* (or exocentric). Another interpretation is that the object is stationary whereas the viewpoint is moved in an opposite way. This type of transformation is called *egocentric*.

In the configuration studied here, the object is displayed on the screen but the screen itself moves during manipulation. We thus do not consider whether the object appears to move from the viewpoint of the user, but how it moves *on the screen* relative to the device. Allocentric mappings differ from egocentric manipulations based on whether the object moves on the screen in the same direction as the device (allocentric) or in the *opposite* direction (egocentric), as illustrated in Figure 5. This notion differs from directional compliance. Directional compliance means that the manipulated object moves along the same axis as the device, as opposed to another direction. Allocentric and egocentric manipulation, however, refer to the *sense* of object motion along this axis. For example, if the object is moving along the same axis as the device but in an opposite way, then the mapping is directionally compliant and egocentric.

Using our formalism from Section 4, we can make any mapping either allocentric or egocentric by simply changing the gain factor k_t. A positive gain factor, or no explicit gain (i. e., a gain factor of 1), makes mappings allocentric. Negative gain factors invert translations and rotations and the mapping becomes egocentric.

Gain factor of −1. This particular value is significant for position control mappings. It causes the direction of mobile device translations and rotations to be reversed, leaving their amplitude unchanged. In other words, a gain factor of −1 applies the exact opposite of the mobile device motion to the manipulated object. The mapping thus becomes egocentric, but remains isomorphic.

This has a notable effect on rotations. On a mobile device, the manipulated object is displayed on the screen, which is rotating along with the device during manipulation. By applying the exact opposite of the screen rotations to the object, these rotations are canceled from the user's point of view. The manipulated object will thus appear to have a *fixed orientation* relative to the real world. Interestingly, the effect is identical in the absolute and relative mappings. For a constant gain factor $k_t=-1\ \forall t>0$, the absolute mapping results in the following object orientations:

$$\begin{aligned} qd_t &= \Delta qc_t^{-1}\,qd_0 \\ &= \left(qc_0^{-1}\,(qc_t\,qc_0^{-1})\,qc_0\right)^{-1}\,qd_0 \\ &= \left(qc_0^{-1}\,(qc_t\,qc_0^{-1})^{-1}\,qc_0\right)\,qd_0 \\ &= \left(qc_0^{-1}\,(qc_0\,qc_t^{-1})\,qc_0\right)\,qd_0 \\ &= (qc_t^{-1}\,qc_0)\,qd_0 \end{aligned}$$

And the relative mapping results in:

$$\begin{aligned} qd_t &= \Delta qc_t^{-1}\,qd_{t-1} \\ &= \left(qc_{t-1}^{-1}\,(qc_t\,qc_{t-1}^{-1})\,qc_{t-1}\right)^{-1}\,qd_{t-1} \\ &= (qc_{t-1}^{-1}\,qc_t)^{-1}\,qd_{t-1} \\ &= (qc_{t-1}^{-1}\,qc_t)^{-1}\,(qc_{t-2}^{-1}\,qc_{t-1})^{-1}\ldots(qc_0^{-1}\,qc_1)^{-1}\,qd_0 \\ &= (qc_t^{-1}\,qc_{t-1})\,(qc_{t-1}^{-1}\,qc_{t-2})\,\ldots\,(qc_1^{-1}\,qc_0)\,qd_0 \\ &= (qc_t^{-1}\,qc_0)\,qd_0 \end{aligned}$$

Hence, rotations become strictly equivalent in absolute and relative mappings for a constant gain factor $k_t=-1$. This implies that they now share the *same spatial compliances*. Rotations in the absolute mapping become directionally compliant, as in the equivalent relative mapping. Rotations in the relative mapping become transitive, as in the equivalent absolute mapping. A gain factor of −1 is thus the only way to have *both* directional compliance and transitivity for rotations in position control mappings.

The same equivalence, however, is not guaranteed for translations. As we showed before, spatial compliances of translations not only depend on the device's position but also on its orientation. In the absolute mapping, a constant gain factor $k_t=-1$ results in the following object positions:

$$\begin{aligned} pd_t &= -\Delta pc_t + pd_0 \\ &= -\left(qc_0^{-1}\,(pc_t-pc_0)\,qc_0\right)+pd_0 \\ &= \left(qc_0^{-1}\,(pc_0-pc_t)\,qc_0\right)+pd_0 \end{aligned}$$

The vector pc_0-pc_t is the opposite of the total device translation. One might expect that applying it to the object would compensate for the device translation, making the object appear to have a fixed position in the real world. This vector, however, has to be converted to screen space. In this mapping, the conversion is done according to the initial device orientation qc_0. The object can thus only appear to be stationary if device orientation remains equal to qc_0. In the relative mapping, a constant gain factor of −1 yields:

$$\begin{aligned} pd_t &= -\Delta pc_t + pd_{t-1} \\ &= -\left(qc_{t-1}^{-1}\,(pc_t-pc_{t-1})\,qc_{t-1}\right)+pd_{t-1} \\ &= \left(qc_{t-1}^{-1}\,(pc_{t-1}-pc_t)\,qc_{t-1}\right)+pd_{t-1} \\ &= \left(qc_{t-1}^{-1}\,(pc_{t-1}-pc_t)\,qc_{t-1}\right) \\ &\quad +\ldots+\left(qc_0^{-1}\,(pc_0-pc_1)\,qc_0\right)+pd_0 \end{aligned}$$

Even though the opposite of each device translation substep $pc_t - pc_{t-1}$ is applied to the manipulated object, each substep is first converted to screen space according to the *intermediate* device orientation. The total object translation thus depends on intermediate device orientations. The object position will only appear to be fixed relative to the real world if device orientation does not change during manipulation, and thus remains equal to qc_0.

Note that simultaneously performing translations and rotations with a gain factor of -1 would therefore *not* result in an AR-like mapping—i. e., the object having both a fixed position and a fixed orientation relative to the real world—at least under the above formulation of the position-control mappings.

5.4 Summary of spatial compliances

Table 1 summarizes spatial compliance properties of each mapping, both for translations and rotations. With $k_t = -1$ we indicate that the property is only guaranteed when the gain factor k_t (see Section 4.6) remained constant and equal to -1 since the start of manipulation.

Table 1: Spatial compliances of each mapping.

	directional compliance		**transitivity**	
	translation	rotation	translation	rotation
absolute	no[a]	$k_t = -1$[b]	yes	yes
relative	yes	yes	no[c]	$k_t = -1$
rate	yes	yes	no	no

a. unless the device orientation is equal to its initial orientation *and* the gain factor is constant

b. regardless of the gain factor, this mapping is still directionally compliant if the device only rotates about a single axis

c. unless there is no device rotation (either its rotation is ignored, or its orientation does not change) *and* the gain factor remains constant

5.5 Choosing between spatial compliances

From the results shown in Table 1, we see that none of the three mappings provide both directional compliance and transitivity in all cases. Furthermore, a choice must be made between allocentric and egocentric manipulation. In this section, we present arguments and evidence to help designers choose between these alternatives.

5.5.1 Directional compliance versus transitivity

We demonstrated above that none of the three main mappings can generally guarantee simultaneous directional compliance and transitivity. According to Table 1, rotations only exhibit both compliances when the mapping is position-controlled, isomorphic, and egocentric ($k_t = -1$). Translations only exhibit both compliances when the mapping is position-controlled, without simultaneous rotations, and with a constant gain factor. Those are substantial constraints which may not be acceptable in many applications. In practice, the choice of the right mapping for a given use case will thus depend on which of the two spatial compliances is the most important.

Directional compliance ensures that the motion of the manipulated object matches the motion applied to the interaction device. According to Bowman et al. (2004), this helps the user anticipate object motion and plan its trajectory during manipulation. There is indeed evidence that directional compliance plays a role in usability and user performance. Fitts and Seeger (1953) showed that user responses were slower and less accurate when visual stimuli and user input were not spatially aligned. Furthermore, the difference in performance could not be fully eliminated by training. Ware and Arsenault (2004) studied the effect of a rotation between input and display reference frames—i. e., a lack of directional compliance—on a 3D object rotation task. Their results showed an strong reduction of performance with large angles of mismatch, though the effect was more limited with smaller mismatches. Van Rhijn and Mulder (2006) showed that performance in a 3D manipulation task (translation and rotation) was best when object motion matches device motion relative to the object reference frame, which corresponds to our description of directional compliance. Otherwise, completion time increased significantly. Directional compliance thus appears to be essential for effective manipulation—unless the device is not rotated much during manipulation so that the misalignment between input and display reference frames would remain small (Poupyrev et al., 2000).

Transitivity, or nulling compliance, is desirable in some situations. As previously mentioned, transitivity is useful to recover from manipulation errors. It allows users to exploit muscle memory (Bowman et al., 2004) to reliably return the object to its initial location (nulling compliance), or any valid intermediate location. Transitivity is also useful when the manipulated object has a meaningful upright orientation such as a human head (Buda, 2012), a building, or a landscape since the object can be easily and predictably returned to an upright orientation. According to Poupyrev et al. (2000), the shape of the interaction device is also important. If the device has a perceivable "base" orientation, a lack of nulling compliance in rotations will be noticed by the user, and may impact usability. Here, the interaction device is the mobile device itself. Most current mobile devices have a planar shape, designed to be facing the user. Therefore, they have a base orientation that can be perceived visually and haptically by the user. The lack of nulling compliance (hence the lack of rotation transitivity) can thus be noticeable during manipulation. Other devices such as cube displays (Stavness et al., 2010) do not have a single preferred orientation. The absence of rotation transitivity might be less noticeable on such devices.

Despite its situational usefulness, transitivity is unfortunately incompatible with directional compliance in most cases as we demonstrated above. At least one study comparing rotation mappings (Buda, 2012) reported that directionally compliant mappings performed better and were rated higher than transitive mappings. This suggests that transitive mappings should be preferred over directionally compliant mappings only for specific applications.

5.5.2 Allocentric versus egocentric mappings

Another key factor for usability is the choice between allocentric and egocentric manipulation. Both alternatives are functionally equivalent: they preserve other spatial compliances and produce equivalent—albeit mirrored—object movements in response to the same device movements. Therefore, a choice cannot be made on this basis. As mentioned in Section 5.3, the difference between allocentric and egocentric manipulation is primarily a question of interpretation (e. g., López et al., 2016). When users manipulate the mobile device, are they expecting to manipulate the object, or the viewpoint on the object? It is important that the mapping matches user expectations (Chan et al., 2003) to reduce manipulation errors and improve usability. We thus need to determine what should be the default setting.

Fitts (1951) introduced the concept of *population stereotype* which refers to the option preferred (or expected) by a majority of users in a given population when faced with an arbitrary choice. A typical example is the fact a pushbutton is expected to be activated when depressed. It should be noted that population stereotypes are defined in relation to a given population, and do not necessarily generalize to users from different cultures or even professions (Wiebe and Vu, 2009). Still, a number of stereotypes were found to be sufficiently prevalent to become design guidelines. For instance, Warrick's principle (Warrick, 1947; Wiebe and Vu, 2009) states that a controlled object should move in the same direction as the side of the control device closest to it. The clockwise-to-increase principle (Wiebe and Vu, 2009) states that a controlled value indicator should increase when the control device is turned in a clockwise direction. However, these guidelines were established for 1D translation tasks with separate control devices. They are thus difficult to apply to 3D manipulation in a locally coupled configuration.

Fewer works have focused on population stereotypes for actual 3D manipulation tasks. Kaminaka and Egli (1985) studied stereotypes for translations and rotations of a cube about each axis. The cube was controlled by a lever which could be pushed forward or pulled backward. Their results suggest that allocentric manipulation might be preferred for some axes, but there were no significant difference in others. Besides, a lever is only a 1D control device whereas a mobile device can be freely moved in 3D space. In another study, Diaz and Sims (2005) investigated "accidental inversions" of rotations, i. e. when a user mistakenly rotates the manipulated 3D object in a direction opposite than intended. The goal was to reveal inconsistencies between what the user was expecting (either allocentric or egocentric manipulation) and the actual mapping encountered. The results revealed two types of user expectations. One group of participants strongly expected a direction that "matched" the motion of the input device, while other participants were more uncertain. At first glance, these results seem to discourage inverted mappings. The meaning of "matching" and "inverted," however, cannot be easily transposed to our case because in this experiment 3D rotations were controlled by a separate 2D input device.

Due to the specifics of the locally coupled configuration it is especially difficult to apply previous results and guidelines to our case. Experiments are generally performed in front of a fixed screen that shows the manipulated object, with a separate input device to control it. In the locally coupled configuration studied here, however, the screen is attached to the input device. The mobile device thus serves a dual function: as a handle to control the object and as a way to depict this object. It can be seen both as an input device controlling the object and as a "window" controlling the viewpoint—the chosen interpretation solely depending on the user.

Therefore, the only results applicable here would be those obtained in a locally coupled configuration, so that users face the two possible interpretations. Unfortunately, previous works on locally coupled mappings do not provide sufficient evidence to decide between allocentric and egocentric mappings. In a tilt-based menu selection interface, Rekimoto (1996) chose to move the menu instead of the cursor (egocentric manipulation) to avoid cases where the cursor would be clamped by the device's screen. Weberg et al. (2001) preferred allocentric cursor manipulation as it "felt very intuitive and natural." In a tilt-based scrolling technique, Bartlett (2000) reported that a number of users expected the image to scroll in one direction and a number of others expected the opposite. In a a tilt-based zoom mapping, Hinckley and Song (2011) merely mentioned that "some users with a different mental model preferred the opposite mapping."

Since there do not appear to be sufficient results applicable to 3D manipulation in a locally coupled configuration, we conducted an exploratory experiment to get more insight into which alternative should be preferred.

Experiment. The goal of this experiment was to identify if there exists a preference for either allocentric of egocentric manipulation—i. e., whether a user expects that the mobile device would control the object or would be moving relative to this object. We assumed that a mapping matching this expectation would be perceived as more *natural*. Therefore, in this experiment we chose a subjective assessment of the "naturalness" of each mode in interactive 3D manipulation tasks.

In addition to this primary question, we also wanted to investigate whether the type of 3D object or the type of virtual scene can influence the preference between allocentric and egocentric manipulation. Whether the object is perceived as "movable" or "static" may influence the preference since it depends on interpretation. Any hint that suggests the viewpoint is moving may also have an influence. We will refer to these cues as *contextual cues*. We thus stated the following hypotheses.

H1: the preferred mode is likely to be the egocentric one when the manipulated object is viewed from inside, since motion of the surrounding environment suggests that the viewpoint is moving;

H2: the preferred mode is likely to be the allocentric one when an object is manipulated relative to a fixed environment (i. e., not moving in screen space) since this would suggest that the viewpoint itself is not moving; and

H3: the preferred mode is likely to be the egocentric one when the manipulated object such as a house is perceived as unmovable since in the real world people tend to move around a house rather than moving the house itself.

To test these hypotheses, we conducted this experiment on four 3D scenes with different contextual cues (Figure 6):

1) a generic object on an empty background,
2) a house model (less likely to be perceived as "movable") on an empty background,
3) the same house model viewed from inside, and
4) a generic object inside the house model.

Procedure and tasks. The experiment proceeded as follows. Participants were first told they would have to "perform translations/rotations by moving/tilting the device". We carefully avoided instructions such as "manipulating the object" or "moving the scene" which could have biased the results. The first two trials were performed with Scene 1, the most generic case. One comprised translation tasks, and the other one rotation tasks, presented in random order. Thus, potential learning effects were minimal for this scene. Participants then performed similar translations and rotation trials with the other scenes. All these remaining trials were presented in random order.

A trial consisted of a series of manipulation tasks, either translations or rotations, in both allocentric and egocentric modes. The main purpose of these tasks was to encourage the use of the interface so that participants could rate each mode in realistic conditions. The canonical task for object manipulation is the *docking task* in which an object must be translated and/or rotated

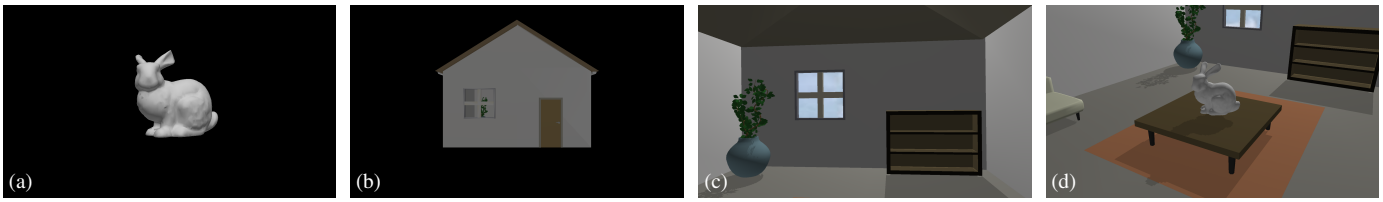

Figure 6: Illustrations of the four scenes used in the experiment: (a)–(d) correspond to scenes 1–4, respectively. Participants were asked to perform translations or rotations, starting from a random location, in order to obtain the result shown above. In scene 4 (d), everything was fixed in screen space apart from the object on the table.

from an initial location to a target location. The object was initially put at a random position or orientation (depending on the type of trial) within the scene. Generally, the target location is visually represented by a "ghost" copy of the object on the screen. However, we could not display the docking target on the screen as we hypothesized (H2) that a fixed object in screen space could influence the results. Instead, we printed on paper an image of the target position/orientation for each scene (Figure 6) and we asked participants to "try to obtain the same result". On success, the object was translated or rotated to a different location for them to try again, as many times as a participant needed to form an opinion on each mode.

There were two unlabeled, randomly assigned buttons that allowed participants to switch freely between allocentric and egocentric modes. In each trial, participants were asked to perform the manipulation task a few times with each mode before rating them. Ratings were given on a Likert-type scale with 4 points (to avoid neutral answers) going from "unnatural" to "natural". The meaning of "natural" was explained as "whether your actions produce a translation or a rotation in the direction you expected".

Apparatus. The mobile device was a 7" tactile tablet, held horizontally with two hands. We chose to implement a relative position control mapping, as described above, with a constant gain factor $k_t = 1$ for translations and $k_t = 3$ for rotations. The direction of translations and rotations was inverted in the egocentric condition. Tracking was accomplished with the integrated gyroscopes and magnetic sensors for orientation as well as with an optically tracked marker behind the tablet for position. We do not expect the chosen mapping or device to have an influence on the answers since the preference between allocentric and egocentric manipulation relates to the user's own mental model. The results should thus remain applicable to other mappings and mobile devices.

Participants. Ten unpaid participants (2 female, 8 male) took part in this experiment. Four of them were familiar with 3D manipulation techniques—though not in this particular configuration—and the others were novices.

The purpose of this study was exploratory: obtaining a first insight into the preferred mode of manipulation in a locally coupled configuration. Although the limited number of participants might not be enough to draw definite conclusions, it should nevertheless allow us to observe sufficiently strong effects.

Results and discussion. The ratings given by participants to each condition are shown in in Figure 7. In addition, we used the Wilcoxon signed-rank test to quantify the difference in ratings between the allocentric and egocentric modes. The effect size r was computed from the z statistic as described by Fritz et al. (2012). Guidelines are that $r > 0.5$ is a large effect, $r > 0.3$ is a medium effect and $r > 0.1$ is a small effect (Fritz et al., 2012). We also report a bootstrapped standard error σ of the effect size from which confidence intervals can be derived for a meta-analysis (bias was <0.01 in all cases).

In Scene 1 which was presented first to minimize learning effects and which did not contain any contextual cues, we had no reason to expect that participants would prefer either allocentric or egocentric manipulation. Yet, the results reveal a noticeable difference between the two modes. With translations, there was a strong preference ($|r|$=0.57, σ=0.08) for egocentric manipulation. With rotations, there was a slightly less contrasted but still strong preference ($|r|$=0.54, σ=0.09) for allocentric manipulation.

The three other scenes were designed to study the influence of contextual cues. Scene 2 was almost identical to Scene 1, except for the manipulated object: a house, i. e. an object that is generally *unmovable* in reality as well as in many virtual environments. This scene was designed to test our hypothesis **H3** that egocentric manipulation is preferred when the manipulated object is perceived as unmovable. With translations, there was a distinct preference for egocentric manipulation ($|r|$=0.47, σ=0.16). The ratings also seem more contrasted than in Scene 1, which may support our hypothesis. With rotations, both allocentric and egocentric manipulation received a similar proportion of positive and negative ratings ($|r|$=0.02, σ=0.22). Compared to Scene 1, however, allocentric rotation was rated lower and egocentric rotation was rated higher. The preference for egocentric rotation was thus higher than that for Scene 1, supporting hypothesis **H3**. Allocentric rotation may also be a sensible choice in this case since the ratings were similar.

Scene 3 was designed to test our hypothesis **H1** that egocentric manipulation is preferred when the manipulated object is viewed from inside. Indeed, the results show a preference for egocentric manipulation with both translations and rotations. This preference was strong for rotations ($|r|$=0.64, σ=0.007). It was less marked but still noticeable for translations ($|r|$=0.42, σ=0.15). These results thus support our hypothesis **H1**, although we cannot explain why this effect was weaker for translations.

Scene 4 was designed to test our hypothesis **H2** that allocentric manipulation is preferred when an object is manipulated within a surrounding environment. Concerning rotations, allocentric manipulation was widely considered as natural which tends to support our hypothesis, although egocentric manipulation was neither truly preferred nor truly rejected by participants. The difference in ratings between the two conditions was $|r|$=0.48 (σ=0.13). Surprisingly, preferences for translations did not match our **H2** hypothesis. Egocentric translations were perceived as more natural than allocentric translations ($|r|$=0.51, σ=0.13). At

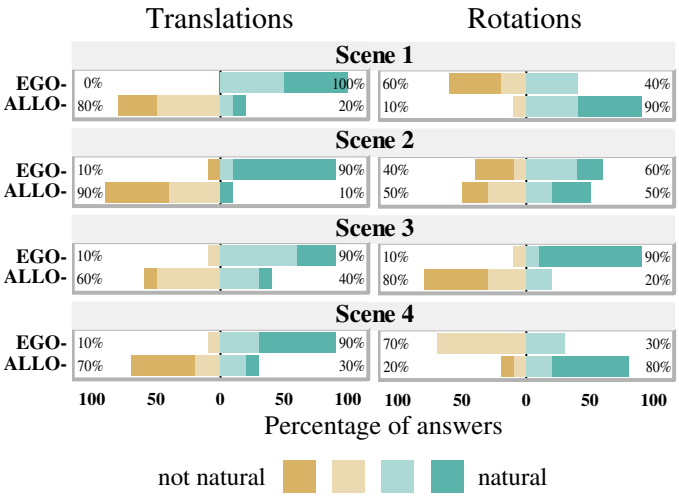

Figure 7: Subjective "naturalness" ratings for allocentric and egocentric mappings in each of the four scenes.

this point, we are left with two possible explanations. Either the contextual cues of the 3D scene actually have little or no influence on translations and thus any scene would lead to the same (egocentric) preference as Scene 1 (some participants commented that they did not pay attention to the surrounding environment in Scene 4 and just performed the task as if it were Scene 1). Or there could be an unexpected bias in the experiment that led users toward the egocentric interpretation. One possible cause might be the marker used for position tracking. Some participants reported that they were consciously moving the device relative to the marker, which may reveal a potential bias. There are also arguments, however, against the existence of such a bias: If one scene was truly encouraging allocentric translation, while a bias in the experiment caused a shift toward egocentric translation, then the results should be closer to a tie between the two modes (i. e., similar to the rotation ratings in Scene 2).

To summarize, for translations the preference was clearly toward egocentric manipulation in every case. It is unknown whether this was caused by an experimental bias or just a coincidence. But if we were to draw guidelines from these preliminary results, we would recommend to make *egocentric translations* the default setting. Concerning rotations, they were apparently affected by the contextual cues in different types of scenes. The preferences were not always as marked as for translations but we can still derive recommendations from these results. Allocentric rotations seem to be a safe choice in most cases, except in Scene 3. We would thus tend to recommend making *allocentric rotations* the default setting, *except* when the manipulated object is actually viewed from inside.

6 Motion Space

Using the motion of a handheld device as input modality is subject to a limiting factor: the space in which the device can be translated and rotated by the user—which we refer to as *motion space*. Depending on the application, the manipulated 3D object may have to be moved across arbitrarily large virtual distances or rotated by arbitrarily large angles. The user, however, cannot apply arbitrarily large displacements to the mobile device itself. In addition, the device itself may not provide enough visual feedback in some positions or orientations. Therefore, a manipulation mapping must provide a way to address these limitations.

In this section, we first present the factors that influence the motion space. We then present several solutions to overcome this limited range in 3D manipulation mappings, with arguments for and against each alternative.

6.1 Factors influencing the motion space

6.1.1 Anatomical constraints

Since the user is holding the mobile device during manipulation, device motion is primarily limited to the space reachable by the user's hands. Translations are limited by arm length[4], and rotations are limited by wrist constraint (Rahman et al., 2009). The exact limitations, however, depend on how the mobile device is held. First, a handheld object can be held with either one or two hands. Two types of grasps can also be distinguished: a *power grasp* (or force grasp) in which the device is held firmly in the hand(s), and a *precision grasp* in which the device is held by the fingers (Fitzmaurice, 1996).

In a power grasp, device movements are more constrained. When the device is held with one hand, translations and rotations are limited by the space reachable by the hand and orientations reachable by the wrist. With two hands, device movements become subject to the limitations of both hands and wrists at the same time, although the range of rotations may increase by using the hands as pivots. In a precision grasp, fingers can compensate for these constraints and greatly enlarge the motion space. With two hands, the freedom of manipulation increases even more.

6.1.2 Visual feedback

For devices whose display surface does not cover all directions, a second restriction to motion space is screen visibility. Manipulation of a 3D virtual object normally requires continuous visual feedback to control the path of the object, so the screen has to remain visible during manipulation. This further reduces the motion space to device configurations that maintain visual feedback. Since many current mobile devices only have a single flat front-facing screen, this restricts the range of possible orientations to those where the screen is facing the user (Rahman et al., 2009). Recent panel technologies (such as IPS or OLED) provide close to 180° viewing angles. Still, a flat one-sided screen—regardless of its technology—necessarily becomes unreadable when facing away from the user.

6.1.3 Comfortable motion space

Even though device motion space can theoretically extend up to arm's length and to wide viewing angles, such positions and orientations are not always practical for prolonged manipulation. Holding the device at arm's length causes fatigue and reduces manipulation accuracy. The screen may also be visible but not necessarily legible. When the device is translated away from the user's eyes, the screen contents appear smaller which reduces legibility. When the device is rotated, perspective distortion (for flat screens) or less defined/darker areas (for some spherical and volumetric displays) can make the screen contents unreadable, effectively interrupting visual feedback. Therefore, we could further distinguish the *full motion space*, theoretically reachable by the user, from the smaller *comfortable motion space* in which the

4. Unless the user moves within tracking space (e. g., by walking while holding the mobile device), a case which we do not consider here.

device can be handled comfortably. A good mapping should not require the user to move the device outside the comfortable motion space. Still, the additional accessible space can be exploited for short-term or less frequent manipulation tasks.

The extents of the full motion space and the comfortable motion space may be determined experimentally. However, the results would be strongly device-specific. Such an experiment thus falls outside the scope of our theoretical framework. A future, dedicated study could be conducted on the most commonly available devices, in order to better adjust the mappings to current mobile platforms.

6.2 Overcoming motion space limitations

There are different ways to work around motion space limitations in a manipulation mapping. Although this remains an open problem, we present here some of the traditional solutions. We then demonstrate how more advanced techniques can be implemented by extending the mappings we presented before.

6.2.1 Rate control

One way to circumvent almost entirely this issue is to use a rate control mapping. Under a rate control mapping, the manipulated object moves continuously as long as the mobile device is away from the starting position and orientation, with a speed that depends on the distance from this point. Thus, with a reasonable gain factor and/or given enough time, unlimited translations and rotations can be applied to the manipulated object while keeping the device well inside its motion space.

Unlimited movement is undoubtedly a significant advantage of rate control mappings. However, such mappings are not without drawbacks compared to position control. For instance, changing the direction of motion *first* requires to return the device to its initial location, *then* to move it toward the new direction. This is slower than position control, in which such change of direction would happen instantly. Stopping the manipulated object at a given position or orientation is also difficult with rate control. This requires to bring back the mobile device to its initial location with a precise timing. Since a mobile device is untethered and freely manipulable, i. e., isotonic, there is no self-centering mechanism and no feedback to help the user return to the initial location (Zhai and Milgram, 1998). The numerous degrees of freedom in 3D space make it particularly difficult. This is supported by experimental results: Zhai (1998) showed that rate control mappings are less efficient than position control mappings, when isotonic input devices are used for a 3D docking task.

The difficulty of zeroing the device in mid air can be alleviated by adding a "deadband" to the mapping. In our formalism, this can be accomplished by setting the gain factor to 0 when the distance (or angle) between the current device location and its initial location is below a given threshold. We first introduce a *dist* function to compute the distance or angle between two device locations at times t_1 and t_2 (here, $(q)_w$ denotes the real part of a quaternion q):

$$\text{dist}(t_1,t_2) = \begin{cases} \|pc_{t_1} - pc_{t_2}\| & \text{(translations)} \\ 2\arccos\big((qc_{t_1}\, qc_{t_2}^{-1})_w\big) & \text{(rotations)} \end{cases} \quad (9)$$

The distance between the current and initial device locations is given by $\text{dist}(t,0)$. A deadband of radius threshold can thus be obtained with the following gain function:

$$d = \text{dist}(t,0)$$
$$\text{gain}(t) = \begin{cases} 0 & \text{when d} < \text{threshold} \\ (d - \text{threshold}) \,/\, d & \text{otherwise} \end{cases}$$

A deadband, however, uses up a valuable part of the motion space, forcing the device into less comfortable positions and orientations. Also a change of motion direction becomes even less efficient.

Therefore, the use of rate control mappings should be considered carefully. Whether unlimited object motion is worth the loss of accuracy and efficiency in manipulation depends on the application, and especially the frequency of long-distance object manipulation.

6.2.2 Clutching

Position control mappings do not allow unlimited object movement, but there are still ways to overcome the motion space limitation. *Clutching* consists in temporarily disengaging the mapping between device and object motion, returning the device to a more convenient location in the motion space, then resuming manipulation from there. This enables arbitrarily large translations and rotations to be applied to the manipulated object, by decomposing them into smaller steps that fit within the motion space. Clutching requires a way to explicitly engage or disengage manipulation, such as a button on the mobile device. In any case, such a mechanism is recommended since it lets users move the device for other purposes than object manipulation.

However, clutching is best avoided *during* manipulation. By interrupting the interaction, it slows down object manipulation and reduces efficiency (Jellinek and Card, 1990). The further the object has to be moved or rotated, the more clutching must be used and the less efficient manipulation becomes. Clutching also causes "wasted" user movement (Zhai and Milgram, 1998). Moving a handheld device in 3D space is demanding, and frequent clutching can quickly lead to user fatigue.

Clutching is generally required for position control mappings to support exceptionally large translations and rotations. Occasional clutching is a normal occurrence in a position control mapping. But a good mapping should be designed to reduce it as much as possible for normal use cases.

6.2.3 Larger gain factor

The need for clutching can be reduced by mapping device movements to larger object movements. In the formalism we presented before, this is easily accomplished by increasing the gain factor k_t (see Section 4.6) to a value greater than 1. With an increased gain factor, the same device movements will result in larger object translations or rotations. The larger the gain factor, the less likely the user will be limited by motion space during manipulation.

The gain factor, on the other hand, cannot be infinite. The need for clutching can thus never be completely eliminated with this method. Furthermore, human motor resolution is also limited (Bérard et al., 2011). There is a limit to how much the gain can be increased before it exceeds a user's ability to control the manipulated object, after which accuracy begins to drop. The exact threshold depends both on the user's own motor skills and the manipulation task to be carried out. Conversely, reducing the gain factor to less than 1 could artificially increase manipulation accuracy beyond the human motor abilities (within the limits of the tracking system). But this would also require larger device

movements to translate or rotate the object along the same distance, and cause more frequent clutching.

Increasing the gain factor is thus an effective solution against clutching in position control mappings, but leads to a trade-off between motion space and accuracy. To avoid this trade-off, more advanced strategies must be employed for managing the gain factor.

6.2.4 Dynamic gain factor

The problem with a static gain factor is that it affects every manipulation task performed with the interface. Ideally, the gain factor should be dynamically adapted to the type of manipulation: larger for coarse long-distance object manipulation, and smaller for precise short-distance manipulation. However, in order to do that, we need to know what the user intends to perform next.

One solution is to give the user explicit control on the gain factor (e. g., Ware and Baxter, 1989). However, this would require an additional input modality, since all the degrees of freedom of the mobile device are already used for manipulation. This would also cause additional cognitive load during interaction (the need for a specific learning period is mentioned by Ware and Baxter (1989)). Another solution is to give the user *implicit* control on the gain factor: by varying the gain factor according to patterns in device movement. We present here two methods to implicitly control the gain factor.

Gain based on distance. This method consists in increasing the gain when the mobile device moves or rotates away from the location where manipulation was initiated. It is motivated by some observations and assumptions. First, users generally begin manipulation from a comfortable device position and orientation. If users only move or rotate the device by a small amount around this starting location, we can assume they are trying to perform precise manipulation of the virtual object. We thus keep the gain factor low in this area. This is safe because clutching is unlikely to be needed when the device is still inside the comfortable motion space. On the other hand, if users move or rotate the device far away from its initial location, we can assume they are attempting to translate or rotate the virtual object by a large amount. We thus increase the gain factor to support long-distance object manipulation and reduce the risk of reaching the limits of the motion space. Since there is no way to know what the user intends to do before the actual movement occurred, we can transition smoothly between the two situations by making the gain a function of the distance from the starting location. This behavior is implemented by the following gain function, based on the *dist* function (9) we introduced before:

$$\text{gain}(t) = a + b\ \text{dist}(t,0)^c$$

Previous works have proposed similar techniques that increase the gain with distance, such as the mapping proposed by Poupyrev et al. (1999) for rotations, or the non-linear part of the the Go-Go Interaction Technique (Poupyrev et al., 1996) for translations. In the gain function we formalize here, the terms a (minimum gain factor), b (scaling coefficient) and c (exponent) can be adjusted to recreate these mappings.

Gain based on speed. A different approach, inspired by the "pointer acceleration" technique for computer mice, is to base the gain factor on the mobile device's *speed*. In other words, the faster the mobile device is moved/rotated by the user, the faster the manipulated object will move/rotate. In the real world, precise object manipulation has to be performed slowly and carefully, whereas coarser and larger-scale manipulation can be performed with faster movements. We can thus exploit this metaphor to let users implicitly control the amount of gain they expect during manipulation.

The speed of the mobile device corresponds to the distance or angle crossed by the device between times $t-1$ and t, divided by the interval of time Δt elapsed between the two steps. Note that this method relies on incremental device translations and rotations, it is thus *only meaningful for relative position control mappings*. On such mappings, the speed-based gain can be implemented with the following function:

$$\text{gain}(t) = a + b\left(\frac{\text{dist}(t,t-1)}{\Delta t}\right)^c$$

As in the previous method, the a, b and c terms make it possible to fine-tune the shape of the gain function, for instance to match existing pointer acceleration schemes.

7 Conclusion

In this work, we investigated a specific mode of interaction: using the motion of a mobile device to manipulate a 3D object displayed on the device itself. More precisely, we conducted the first in-depth analysis of the possible *mappings* between device motion and object motion in this particular configuration. We introduced a formalization of the main mappings, which—unlike many previous works—covers both device translations and device rotations and supports a variable control-display gain. We performed a theoretical analysis of the spatial compliance properties of each mapping. We then reviewed the arguments for each property, contributing new results when needed, allowing implementors to choose which mapping is best suited to their needs. We conducted a first study on user preference between allocentric and egocentric manipulation, in order to help implementors select a sensible default setting. We finally presented a theoretical analysis of the motion space available for manipulation while holding a mobile device, as well as several solutions including non-isomorphic gain functions to overcome this limitation.

By presenting the main mappings in a consistent notation and addressing important questions often overlooked by previous work, our framework provides implementors with readily available advice to design such manipulation techniques. Furthermore, our framework is applicable to any handheld device capable of displaying a virtual object (according to the definition given in Section 3), regardless of its shape or the technologies used. Since mobile devices are going to become even more ubiquitous, our work may serve a basis for 3D manipulation techniques on future devices.

To better understand the further implications of this mode of interaction, a number of user studies could be conducted in the future. One study could compare the usability of the non-isomorphic gain functions presented in Section 6. Although they are based on real-world metaphors, it is unknown whether they can be easily understood in the specific configuration studied here. Another question of interest is the usability of integrated (i. e., simultaneous) translations and rotations. Our formalized mappings cover both translations and rotations, and our analysis of spatial compliances did take into account the effects of simultaneously rotating and translating the *device*. However, the usability consequences of simultaneously translating and rotating the *manipulated object* remain to be studied. In addition, different mappings may be preferable for translations than for rotations (e. g., relative position control for translations and rate control for rotations). It might also be preferable to use different gain functions. Further work is

thus needed to evaluate the effects of having different mappings for translations and rotations, and to identify potential usability challenges arising in this new situation.

REFERENCES

Alexander, J., Lucero, A., and Subramanian, S. (2012). Tilt displays: Designing display surfaces with multi-axis tilting and actuation. In *Proc. MobileHCI*, pages 161–170. ACM, New York. doi: 10.1145/2371574.2371600

Bade, R., Ritter, F., and Preim, B. (2005). Usability comparison of mouse-based interaction techniques for predictable 3D rotation. In *Proc. Smart Graphics*, pages 138–150. Springer, Berlin/ Heidelberg. doi: 10.1007/11536482_12

Bartlett, J. (2000). Rock 'n' Scroll is here to stay. *IEEE Computer Graphics and Applications*, 20(3):40–45. doi: 10.1109/38.844371

Benko, H., Wilson, A. D., and Balakrishnan, R. (2008). Sphere: Multi-touch interactions on a spherical display. In *Proc. UIST*, pages 77–86. ACM, New York. doi: 10.1145/1449715.1449729

Benzina, A., Dey, A., Tönnis, M., and Klinker, G. (2012). Empirical evaluation of mapping functions for navigation in virtual reality using phones with integrated sensors. In *Proc. APCHI*, pages 149–158. ACM, New York. doi: 10.1145/2350046.2350078

Bérard, F., Wang, G., and Cooperstock, J. R. (2011). On the limits of the human motor control precision: The search for a device's human resolution. In *Proc. INTERACT*, pages 107–122. Springer, Berlin/Heidelberg. doi: 10.1007/978-3-642-23771-3_10

Bowman, D. A., Kruijff, E., LaViola, Jr., J. J., and Poupyrev, I. (2004). *3D User Interfaces: Theory and Practice*. Addison-Wesley, Boston.

Britton, E. G., Lipscomb, J. S., and Pique, M. E. (1978). Making nested rotations convenient for the user. *ACM SIGGRAPH Computer Graphics*, 12(3):222–227. doi: 10.1145/800248.807394

Buda, V. (2012). Rotation techniques for 3D object interaction on mobile devices. Master's thesis, Utrecht University, the Netherlands.

Chan, A. H. S., Shum, V. W. Y., Law, H. W., and Hui, I. K. (2003). Precise effects of control position, indicator type, and scale side on human performance. *The International Journal of Advanced Manufacturing Technology*, 22(5–6):380–386. doi: 10.1007/s00170-002-1491-z

Cho, S.-J., Choi, C., Sung, Y., Lee, K., Kim, Y.-B., and Murray-Smith, R. (2007). Dynamics of tilt-based browsing on mobile devices. In *CHI Extended Abstracts*, pages 1947–1952. ACM, New York. doi: 10.1145/1240866.1240930

Daiber, F., Li, L., and Krüger, A. (2012). Designing gestures for mobile 3D gaming. In *Proc. MUM*, pages 3:1–3:4. ACM, New York. doi: 10.1145/2406367.2406371

Diaz, D. D. and Sims, V. K. (2005). Accidental inversion during three-dimensional orientational control. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 49(13):1248–1250. doi: 10.1177/154193120504901307

Du, Y., Ren, H., Pan, G., and Li, S. (2011). Tilt & touch: Mobile phone for 3D interaction. In *Proc. UbiComp*, pages 485–486. ACM, New York. doi: 10.1145/2030112.2030183

Eslambolchilar, P. and Murray-Smith, R. (2008). Control centric approach in designing scrolling and zooming user interfaces. *International Journal of Human-Computer Studies*, 66(12):838–856. doi: 10.1016/j.ijhcs.2008.07.005

Feinstein, D. (2002). View navigation and magnification of a hand-held device with a display. US Patent No. 6,466,198.

Fitts, P. M. (1951). Engineering psychology and equipment design. In Stevens, S. S., editor, *Handbook of Experimental Psychology*, pages 1287–1340. Wiley.

Fitts, P. M. and Seeger, C. M. (1953). SR compatibility: Spatial characteristics of stimulus and response codes. *Journal of Experimental Psychology*, 46(3):199–210. doi: 10.1037/h0062827

Fitzmaurice, G. W. (1996). *Graspable user interfaces*. PhD thesis, University of Toronto, Canada.

Fitzmaurice, G. W., Zhai, S., and Chignell, M. H. (1993). Virtual reality for palmtop computers. *ACM Transactions on Information Systems*, 11(3):197–218. doi: 10.1145/159161.159160

Fritz, C. O., Morris, P. E., and Richler, J. J. (2012). Effect size estimates: Current use, calculations, and interpretation. *Journal of Experimental Psychology: General*, 141(1):2–18. doi: 10.1037/a0024338

Fröhlich, B., Plate, J., Wind, J., Wesche, G., and Göbel, M. (2000). Cubic-Mouse-based interaction in virtual environments. *IEEE Computer Graphics and Applications*, 20(4):12–15. doi: 10.1109/38.851743

Grossman, T. and Balakrishnan, R. (2006). The design and evaluation of selection techniques for 3D volumetric displays. In *Proc. UIST*, pages 3–12. ACM, New York. doi: 10.1145/1166253.1166257

Ha, T. and Woo, W. (2011). ARWand: Phone-based 3D object manipulation in augmented reality environment. In *Proc. ISUVR*, pages 44–47. IEEE Computer Society, Los Alamitos. doi: /10.1109/ISUVR.2011.14

Ha, T. and Woo, W. (2013). Poster: A pilot study on stepwise 6-DoF manipulation of virtual 3D objects using smartphone in wearable augmented reality environment. In *Proc. 3DUI*, pages 137–138. IEEE Computer Society, Los Alamitos. doi: 10.1109/3DUI.2013.6550216

Hansen, T. R., Eriksson, E., and Lykke-Olesen, A. (2006). Mixed interaction space – Expanding the interaction space with mobile devices. In *People and Computers XIX — The Bigger Picture*, pages 365–380. Springer, London. doi: 10.1007/1-84628-249-7_23

Harrison, B. L., Fishkin, K. P., Gujar, A., Mochon, C., and Want, R. (1998). Squeeze me, hold me, tilt me! An exploration of manipulative user interfaces. In *Proc. CHI*, pages 17–24. ACM, New York. doi: 10.1145/274644.274647

Henrysson, A., Billinghurst, M., and Ollila, M. (2005). Virtual object manipulation using a mobile phone. In *Proc. ICAT*, pages 164–171. ACM, New York. doi: 10.1145/1152399.1152430

Hinckley, K., Pausch, R., Goble, J. C., and Kassell, N. F. (1994). Passive real-world interface props for neurosurgical visualization. In *Proc. CHI*, pages 452–458. ACM, New York. doi: 10.1145/191666.191821

Hinckley, K., Pierce, J., Sinclair, M., and Horvitz, E. (2000). Sensing techniques for mobile interaction. In *Proc. UIST*, pages 91–100. ACM, New York. doi: 10.1145/354401.354417

Hinckley, K. and Song, H. (2011). Sensor synaesthesia: Touch in motion, and motion in touch. In *Proc. CHI*, pages 801–810. ACM, New York. doi: 10.1145/1978942.1979059

Ishii, H. (2008). The tangible user interface and its evolution. *Communications of the ACM*, 51(6):32–36. doi: 10.1145/1349026.

1349034

Issartel, P., Guéniat, F., and Ammi, M. (2014). A portable interface for tangible exploration of volumetric data. In *Proc. VRST*, pages 209–210. ACM, New York. doi: 10.1145/2671015.2671130

Jellinek, H. D. and Card, S. K. (1990). Powermice and user performance. In *Proc. CHI*, pages 213–220. ACM, New York. doi: 10.1145/97243.97276

Joshi, N., Kar, A., and Cohen, M. (2012). Looking at you: Fused gyro and face tracking for viewing large imagery on mobile devices. In *Proc. CHI*, pages 2211–2220. ACM, New York. doi: 10.1145/2207676.2208375

Kaminaka, M. S. and Egli, E. A. (1985). Lever controls on specialised farm equipment: Some control/response stereotypes. *Applied Ergonomics*, 16(3):193–199. doi: 10.1016/0003-6870(85)90008-0

Katzakis, N. and Hori, M. (2009). Mobile phones as 3-DOF controllers: A comparative study. In *Proc. DASC*, pages 345–349. IEEE Computer Society, Los Alamitos. doi: 10.1109/DASC.2009.76

Klatzky, R. L. (1998). Allocentric and egocentric spatial representations: Definitions, distinctions, and interconnections. In *Spatial Cognition*, pages 1–17. Springer, Berlin/Heidelberg. doi: 10.1007/3-540-69342-4_1

Kratz, S. and Rohs, M. (2010). Extending the virtual trackball metaphor to rear touch input. In *Proc. 3DUI*, pages 111–114. IEEE Computer Society, Los Alamitos. doi: 10.1109/3DUI.2010.5444712

LaViola, J. J. and Katzourin, M. (2007). An exploration of non-isomorphic 3D rotation in surround screen virtual environments. In *Proc. 3DUI*, pages 49–54. IEEE Computer Society, Los Alamitos. doi: 10.1109/3DUI.2007.340774

Liang, R.-H. (2013). Augmenting the input space of portable displays using add-on Hall-sensor grid. In *UIST Adjunct Proc.*, pages 33–36. ACM, New York. doi: 10.1145/2508468.2508470

López, D., Oehlberg, L., Doger, C., and Isenberg, T. (2016). Towards an understanding of mobile touch navigation in a stereoscopic viewing environment for 3D data exploration. *IEEE Transactions on Visualization and Computer Graphics*, 22. To appear. doi: 10.1109/TVCG.2015.2440233

Lopez-Gulliver, R., Yoshida, S., Yano, S., and Inoue, N. (2009). gCubik: Real-time integral image rendering for a cubic 3D display. In *SIGGRAPH Emerging Technologies*, page 11:1. ACM, New York. doi: 10.1145/1597956.1597967

Lyons, K., Nguyen, D., Ashbrook, D., and White, S. (2012). Facet: A multi-segment wrist worn system. In *Proc. UIST*, pages 123–130. ACM, New York. doi: 10.1145/2380116.2380134

Marzo, A., Bossavit, B., and Hachet, M. (2014). Combining multi-touch input and device movement for 3D manipulations in mobile augmented reality environments. In *Proc. SUI*, pages 13–16. ACM, New York. doi: 10.1145/2659766.2659775

Massimino, M. J., Sheridan, T. B., and Roseborough, J. B. (1989). One handed tracking in six degrees of freedom. In *Proc. Systems, Man and Cybernetics*, volume 2, pages 498–503. IEEE, New York. doi: 10.1109/ICSMC.1989.71346

Mossel, A., Venditti, B., and Kaufmann, H. (2013). 3DTouch and HOMER-S: Intuitive manipulation techniques for one-handed handheld augmented reality. In *Proc. VRIC*, pages 12:1–12:10. ACM, New York. doi: 10.1145/2466816.2466829

Neale, S., Chinthammit, W., Lueg, C., and Nixon, P. (2013). RelicPad: A hands-on, mobile approach to collaborative exploration of virtual museum artifacts. In *Proc. INTERACT*, pages 86–103. Springer, Berlin/Heidelberg. doi: 10.1007/978-3-642-40483-2_7

Noma, H., Miyasato, T., and Kishino, F. (1996). A palmtop display for dextrous manipulation with haptic sensation. In *Proc. CHI*, pages 126–133. ACM, New York. doi: 10.1145/238386.238454

Oakley, I. and O'Modhrain, S. (2005). Tilt to scroll: Evaluating a motion based vibrotactile mobile interface. In *Proc. World Haptics*, pages 40–49. IEEE Computer Society, Los Alamitos. doi: 10.1109/WHC.2005.138

Poupyrev, I., Billinghurst, M., Weghorst, S., and Ichikawa, T. (1996). The Go-Go interaction technique: Non-linear mapping for direct manipulation in VR. In *Proc. UIST*, pages 79–80. ACM, New York. doi: 10.1145/237091.237102

Poupyrev, I., Weghorst, S., and Fels, S. (2000). Non-isomorphic 3D rotational techniques. In *Proc. CHI*, pages 540–547. ACM, New York. doi: 10.1145/332040.332497

Poupyrev, I., Weghorst, S., Otsuka, T., and Ichikawa, T. (1999). Amplifying spatial rotations in 3D interfaces. In *CHI Extended Abstracts*, pages 256–257. ACM, New York. doi: 10.1145/632716.632874

Rahman, M., Gustafson, S., Irani, P., and Subramanian, S. (2009). Tilt techniques: Investigating the dexterity of wrist-based input. In *Proc. CHI*, pages 1943–1952. ACM, New York. doi: 10.1145/1518701.1518997

Ramakers, R., Schöning, J., and Luyten, K. (2014). Paddle: Highly deformable mobile devices with physical controls. In *Proc. CHI*, pages 2569–2578. ACM, New York. doi: 10.1145/2556288.2557340

Rekimoto, J. (1996). Tilting operations for small screen interfaces. In *Proc. UIST*, pages 167–168. ACM, New York. doi: 10.1145/237091.237115

Shoemake, K. (1985). Animating rotation with quaternion curves. *ACM SIGGRAPH Computer Graphics*, 19(3):245–254. doi: 10.1145/325165.325242

Small, D. and Ishii, H. (1997). Design of spatially aware graspable displays. In *CHI Extended Abstracts*, pages 367–368. ACM, New York. doi: 10.1145/1120212.1120437

Smith, T. J. and Smith, K. U. (1987). Feedback-control mechanisms of human behavior. In Salvendy, G., editor, *Handbook of Human Factors*, pages 251–293. Wiley, New York.

Song, P., Goh, W. B., Fu, C.-W., Meng, Q., and Heng, P.-A. (2011). WYSIWYF: Exploring and annotating volume data with a tangible handheld device. In *Proc. CHI*, pages 1333–1342. ACM, New York. doi: 10.1145/1978942.1979140

Spindler, M., Büschel, W., and Dachselt, R. (2012). Use your head: Tangible windows for 3D information spaces in a tabletop environment. In *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces*, ITS '12, page 245–254. ACM, New York, NY, USA. doi: 10.1145/2396636.2396674

Spindler, M., Schuessler, M., Martsch, M., and Dachselt, R. (2014). Pinch-drag-flick vs. spatial input: Rethinking zoom & pan on mobile displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, page 1113–1122. ACM, New York, NY, USA. doi: 10.1145/2556288.2557028

Stavness, I., Lam, B., and Fels, S. (2010). pCubee: A perspective-corrected handheld cubic display. In *Proc. CHI*, pages 1381–1390. ACM, New York. doi: 10.1145/1753326.1753535

Teather, R. J. and MacKenzie, I. S. (2014). Position vs. velocity control for tilt-based interaction. In *Proc. GI*, pages 51–58.

CIPS, Toronto.

Tsandilas, T., Dubois, E., and Raynal, M. (2013). Modeless pointing with low-precision wrist movements. In *Proc. INTERACT*, pages 494–511. Springer, Berlin/Heidelberg. doi: 10.1007/978-3-642-40477-1_31

Tsang, M., Fitzmaurice, G. W., Kurtenbach, G., Khan, A., and Buxton, B. (2002). Boom chameleon: Simultaneous capture of 3D viewpoint, voice and gesture annotations on a spatially-aware display. In *Proc. UIST*, pages 111–120. ACM, New York. doi: 10.1145/571985.572001

van Rhijn, A. and Mulder, J. D. (2006). Spatial input device structure and bimanual object manipulation in virtual environments. In *Proc. VRST*, pages 51–60. ACM, New York. doi: 10.1145/1180495.1180507

Wang, J., Zhai, S., and Canny, J. (2006). Camera phone based motion sensing: Interaction techniques, applications and performance study. In *Proc. UIST*, pages 101–110. ACM, New York. doi: 10.1145/1166253.1166270

Ware, C. and Arsenault, R. (2004). Frames of reference in virtual object rotation. In *Proc. APGV*, pages 135–141. ACM, New York. doi: 10.1145/1012551.1012576

Ware, C. and Baxter, C. (1989). Bat Brushes: On the uses of six position and orientation parameters in a paint program. In *Proc. CHI*, pages 155–160. ACM, New York. doi: 10.1145/67449.67482

Warrick, M. J. (1947). Direction of movement in the use of control knobs to position visual indicators. *Psychological Research on Equipment Design*, pages 137–146.

Weberg, L., Brange, T., and Wendelbo-Hansson, Å. (2001). A piece of butter on the PDA display. In *CHI Extended Abstracts*, pages 435–436. ACM, New York. doi: 10.1145/634067.634320

Wiebe, J. and Vu, K.-P. L. (2009). Application of population stereotypes to computerized tasks. In *Proc. Human Interface and the Management of Information*, pages 718–725. Springer, Berlin/Heidelberg. doi: 10.1007/978-3-642-02556-3_81

Yee, K.-P. (2003). Peephole displays: Pen interaction on spatially aware handheld computers. In *Proc. CHI*, pages 1–8. ACM, New York. doi: 10.1145/642611.642613

Zhai, S. (1995). *Human performance in six degree of freedom input control*. PhD thesis, University of Toronto, Canada.

Zhai, S. (1998). User performance in relation to 3D input device design. *ACM SIGGRAPH Computer Graphics*, 32(4):50–54. doi: 10.1145/307710.307728

Zhai, S. and Milgram, P. (1998). Quantifying coordination in multiple DOF movement and its application to evaluating 6 DOF input devices. In *Proc. CHI*, pages 320–327. ACM, New York. doi: 10.1145/274644.274689

APPENDIX: PSEUDOCODE

Code 1 Absolute position control mapping

```
1: INIT()
2: loop
3:     v ← tracker_pos() − pc_0
4:     r ← tracker_rot() × inv(qc_0)
5:     (v, r) ← TRANSFORM(v, r, 0, t)
6:     object_pos ← v + pd_0
7:     object_rot ← r × qd_0
8:     t ← t + 1
9: end loop
```

Code 2 Relative position control mapping

```
1: INIT()
2: loop
3:     v ← pc_t − pc_{t−1}
4:     r ← qc_t × inv(qc_{t−1})
5:     (v, r) ← TRANSFORM(v, r, t, t−1)
6:     object_pos ← v + object_pos
7:     object_rot ← r × object_rot
8:     pc_t ← tracker_pos()
9:     qc_t ← tracker_rot()
10:    t ← t + 1
11: end loop
```

Code 3 Rate control mapping

```
1: INIT()
2: loop
3:     v ← pc_t − pc_0
4:     r ← qc_t × inv(qc_0)
5:     (v, r) ← TRANSFORM(v, r, t, 0)
6:     object_pos ← v + object_pos
7:     object_rot ← r × object_rot
8:     pc_t ← tracker_pos()
9:     qc_t ← tracker_rot()
10:    t ← t + 1
11: end loop
```

Code 4 Common code

```
1: procedure INIT()
2:     t ← 0
3:     pc_0 ← tracker_pos()
4:     qc_0 ← tracker_rot()
5:     pd_0 ← object_pos
6:     qd_0 ← object_rot
7:     t ← t + 1
8: end procedure
9: function TRANSFORM(v, r, t, from_t)
10:    ▷ Conversion to screen space
11:    q ← qc_{from_t}
12:    v ← (inv(q) × quat(v_x, v_y, v_z, 0) × q)_xyz
13:    r ← inv(q) × r × q
14:    ▷ Control-Display gain
15:    k_t ← gain(t)
16:    v ← k_t × v
17:    r ← slerp(identity, r, k_t)
18:    return (v, r)
19: end function
```