HandyCast: Phone-based Bimanual Input for Virtual Reality in Mobile and Space-Constrained Settings via Pose-and-Touch Transfer

Mohamed Kari
Department of Computer Science
ETH Zürich, Switzerland

Christian Holz
Department of Computer Science
ETH Zürich, Switzerland

Figure 1: HandyCast is a scene-agnostic input technique for bimanual and full-range control in expansive virtual environments under physical space constraints, using touch and motion of a hand-held smartphone. HandyCast enables users to (a) operate rich Virtual Reality environments as in Job Simulator with two virtual hands (b) by sensing embodied interaction from as little physical space as a car's passenger seat as users move, turn, and touch the phone like a controller from the comfort of their lap. (c) HandyCast implements a *pose-and-touch transfer function* that individually fuses and amplifies phone and touch motions into position, rotation, and selection parameters for either virtual hand. Building on visual-inertial odometry to retrieve the 6D phone pose, HandyCast brings the 3D input control common in stationary setups with two hand-held controllers to *mobile settings*, using just a headset and a smartphone anywhere without the need for external tracking.

ABSTRACT

Despite the potential of Virtual Reality as the next computing platform for general purposes, current systems are tailored to stationary settings to support expansive interaction in mid-air. However, in mobile scenarios, the physical constraints of the space surrounding the user may be prohibitively small for spatial interaction in VR with classical controllers. In this paper, we present HandyCast, a smartphone-based input technique that enables full-range 3D input with two virtual hands in VR while requiring little physical space, allowing users to operate large virtual environments in mobile settings. HandyCast defines a pose-and-touch transfer function that fuses the phone's position and orientation with touch input to derive two individual 3D hand positions. Holding their phone like a gamepad, users can thus move and turn it to independently control their virtual hands. Touch input using the thumbs fine-tunes the respective virtual hand position and controls object selection. We evaluated HandyCast in three studies, comparing its performance with that of Go-Go, a classic bimanual controller technique. In our open-space study, participants required significantly less physical motion using HandyCast with no decrease in completion time or body ownership. In our space-constrained study, participants achieved significantly faster completion times, smaller interaction volumes, and shorter path lengths with HandyCast compared to Go-Go. In our technical evaluation, HandyCast's fully standalone inside-out 6D tracking performance again incurred no decrease in completion time compared to an outside-in tracking baseline.

CCS CONCEPTS

• **Human-centered computing** → **Virtual reality**; **Interaction techniques**.

KEYWORDS

Virtual reality; VR input; interaction techniques; 3D controller; bimanual interaction.

ACM Reference Format:
Mohamed Kari and Christian Holz. 2023. HandyCast: Phone-based Bimanual Input for Virtual Reality in Mobile and Space-Constrained Settings via Pose-and-Touch Transfer. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3544548.3580677

1 INTRODUCTION

The sinking cost of head-mounted displays is leading to increased adoption of Virtual Reality (VR) with consumers. Experiences range from gaming [51] and fitness [13] to entertainment [2] and virtual

CHI '23, April 23–28, 2023, Hamburg, Germany

ACM ISBN 978-1-4503-9421-5/23/04…$15.00
https://doi.org/10.1145/3544548.3580677

sightseeing [1]. Several productivity applications have also emerged for VR [45, 50], such as 3D modeling and sketching [5].

The uptake of recent VR experiences can also be attributed to the fact that these systems integrate tracking, computation, and interaction all inside just the headset and two hand-held controllers. This makes them portable and suitable outside of controlled home and office environments, providing experiences during parts of the day that offer less entertainment, such as travel and commute.

However, VR scenarios are typically designed for standing interaction in large environments—virtual as well as physical—utilizing the obstacle-free space around the user for input. Thus, while the form factor of VR systems themselves—headsets and controllers—supports mobile operation, not all mobile scenarios support VR use, especially those that constrain the user's space. Some of these may be particularly interesting for VR use, such as in the passenger seat of a car or while traveling on a bus, train, or plane, or simply while waiting in a public space. Such situations offer plenty of time to enjoy virtual experiences, often when seated, yet little space to interact inside them and to perform the required physical motions.

In this paper, we introduce an interaction technique that retains the immersive spatial 3D interaction around the user while minimizing the demands on physical space. We replace the two VR controllers with a ubiquitous substitute—the personal smartphone—and present *HandyCast*, a smartphone-based input technique to control both virtual hands. HandyCast supports quick and full-range interaction with two-hand control in expansive virtual environments through the inertial and optical sensors inside a phone when held like a gamepad. We disambiguate control over the individual virtual left and right hand from complementary thumb-based touch input.

A single smartphone to control two hands in VR

Figure 1a shows a user playing Job Simulator [35], wearing a headset and interacting through his smartphone using HandyCast. While the right virtual hand is interacting far away and the left is operating at medium distance, (b) the user is physically sitting in the passenger seat of a car, controlling both virtual hands in mid-air while resting his physical arms in his lap. HandyCast redirects all motion and touch input on the phone through its (c) *pose-and-touch transfer function* that computes the position, rotation, and manipulation parameters of either virtual hand inside the virtual environment. The user can reach close-by, medium, and distant objects, as HandyCast amplifies phone motions: We smoothly and instantly translate small phone rotations and movements to larger rotations and movements of both virtual hands.

While controlling both hands with only a single controller removes the independence between the two hands, HandyCast accepts touch input from the left and right thumb as a complement to adjust the positions of the virtual hands. This allows users to embody input through *simultaneous* phone movements and touch motions, resulting in *positional bimanuality* where users can individually control each hand's position. HandyCast thereby builds on users' propensity to unwittingly move physical controllers in video games and, thus, additionally embody their intention through body motions, even though such controller motion yields no effect. HandyCast also leverages users' decade-long experience with touchscreens by evaluating their fine-grained touch motions for accurate 3D cursor control in VR.

Though different from bimanual manipulation with simultaneous rotational and positional control for both hands, HandyCast's positional bimanuality affords users the wide variety of interactions required to operate immersive environments as shown in Figure 2.

HandyCast comprises a SteamVR driver that substitutes handheld controller input with the output of HandyCast's pose-and-touch transfer function. This allows spatially operating any VR application through 3D interaction for selection, placement, and tracing, such as pulling levers, operating sliders, or opening doors—all without the need for external tracking hardware or additional VR controllers. Our video demonstrates these in detail.

We first detail the design rationale of our HandyCast technique and present its three-fold evaluation: 1) Our first user study *in a full-range, seated setup* with 12 participants compared HandyCast, the two-controller baseline Go-Go [47], and the smartphone-based 3D cursor technique Tiltcasting [46] in a unimanual and bimanual task. An HTC VIVE system tracked all spatial input to focus our analysis on the technique-specific differences in task completion. We found that participants completed tasks using HandyCast with no significant difference in completion time compared to the two-controller Go-Go technique, but that HandyCast required significantly less physical travel and control space than the two baselines. Participants reported comparable levels of body ownership. 2) Our second user study *in a space-constrained, seated setup* with 20 participants compared HandyCast with an adapted Space-Constrained Go-Go that uses a level of amplification commensurate to fit the constrained space. We found that HandyCast allowed participants to significantly faster select and place objects than Space-Constrained Go-Go while still requiring significantly less space and mid-air travel. 3) Lastly, our *tracking evaluation* measured the impact of the tracking system on task performance. Four new participants completed this study using HandyCast and repeating the input tasks under outside-in (HTC VIVE) 6D tracking (as used in Studies 1 and 2) and inside-out (phone-only) 6D tracking conditions.

Contributions

With the work in this paper, we contribute

- HandyCast, a *pose-and-touch transfer function* that we designed for individual control over two virtual hands through input on a single smartphone in space-constrained settings, allowing significantly faster object selection and placement in space-constrained settings than the two-controller Go-Go technique [47],
- a first *interaction user study in a open-space seated setup* to assess completion time, motion paths, and control space of HandyCast compared to a controller-based and a smartphone-based baseline under technique-optimal conditions,
- a second *interaction user study in a space-constrained seated setup* to assess completion time, motion paths, and control space of HandyCast compared to a space-constrained implementation of the two-controller Go-Go baseline,
- a *tracking evaluation* that investigated the drift incurred by HandyCast's inside-out 6D tracking (phone-only) compared to ideal conditions using outside-in tracking (HTC VIVE),

- a SteamVR *controller driver* that brings HandyCast's scene-agnostic quick and reliable bimanual interaction for operating large VR scenarios inside mobile and possibly space-constrained settings.

2 RELATED WORK

2.1 Interaction metaphors in VR

Manipulation in 3D user interfaces generally comprises selecting, positioning, rotating, and scaling objects [22, 24, 36, 42, 55]. Several decades of research have brought forward a multitude of input techniques for each of these tasks [12, 47] as well as taxonomies [4, 9, 41, 48] and overviews of design parameters [28, 36]. VR UIs are part of a subset of 3D interaction techniques where the user's body is colocated with the virtual environment.

Two main metaphors exist for manipulating objects in VR [48]. Using the *virtual pointer* metaphor, users can select distant objects by pointing at them [48]. Ray casting faces depth ambiguity, which can be challenging in scenes with densely populated and occluded targets, in turn leading to problem-specific variations [22, 37]. In contrast, the *virtual hand* metaphor [48] provides the user with one, two, or more [49] hand representations that mimic physical hand movement to manipulate objects. In its simplest form, physical and virtual hands are colocated. However, since this rigid coupling limits the reach in VR, researchers have explored techniques to amplify hand positions. For example, Go-Go applies non-linear amplification of hand positions to extend the virtual reach far beyond arm's length [47]. Reach-bounded non-linear (RNL) amplification improves ergonomics while maintaining body ownership [56], exploiting dominance of vision over proprioception [6, 17, 19, 43, 52]. At the core of these amplification techniques are transfer functions that map the position of the physical hands to the virtual space. HandyCast builds on these amplification techniques for pose transfer to provide initial hand positions before we then apply touch transfer to obtain the final hand positions.

2.2 Metaphor combinations and extensions

Both metaphors are combined in HOMER [12], which uses a pointer for selection and a hand metaphor for positioning and rotating. Today's commercial VR games typically use hand avatars to power the main gameplay except for targeting tasks, such as shooting. Ray casting is often used for menus (e.g., Resident Evil 4, Warplanes: WW1 Fighters, Half Life: Alyx), except for fully hand-centric games [35] that use virtual hands for menus, too. HandyCast follows the virtual hand metaphor, as pose transfer allows rapid selection of even distant targets, sharing some of the benefits of rays.

Recent research has also studied virtual hand manipulation in VR under specific conditions. Hayatpur et al. investigated how gestural input can be used to specify shape constraints for object manipulation in VR [27]. Yamagami et al. demonstrated how unimanual input can be mapped to bimanual interactions for users who have full use of only one hand [58]. HandyCast builds on this to use a single smartphone for bimanual interaction in VR, especially when augmented with touch input for additional freedom.

Besides controllers, recent projects have leveraged hand pose estimation to define novel transfer functions for object interaction. Force Push detects gestures as input to translate virtual objects [59]. By using a smartphone rather than mid-air hand tracking [23] for input, HandyCast *circumvents classical technical limits* such as noisy estimates, the need for exaggerated gestures irreconcilable with the goal of subtle input, and tracking losses at the edge of the field of view. Additionally, HandyCast *enables novel interaction capabilities* by supporting thumb input, either for refinement or amplification of virtual hand motions. We also incorporate phone-specific affordances such as haptic buttons (e.g., volume buttons) and precise touch gestures.

2.3 Smartphone-based controllers

Researchers have often utilized smartphones and their sensors for mediating input to remote screens (e.g., [7, 10, 11, 21, 30–32]) or interaction with spatial AR projections [25, 26]. For VR, researchers have also investigated substituting traditional controllers with phones (e.g., [57], Phonetroller [39], Handymenu [38]) or tablets [18, 53]. Dias et al. proposed a technique to point at objects using headset gaze and using touch on the phone for selection [18], such as for menu interaction in VR. Chen et al. presented two techniques to use smartphone touch in AR with a cursor with 2 degrees of freedom, placed on rigid walls [16]. TMMD maps the pose of an externally tracked phone isomorphically into the virtual space [57], allowing users to select objects either through ray casting or by walking up to the objects and using touch input to attach to them. Touch gestures on the phone then allow translating and rotating the object. HandyCast builds on this notion of interaction, but differs in three regards as it is 1) designed for seated and low-effort interaction, thus using amplification, 2) optimized for simultaneous control over *both* virtual hands, and 3) aimed at maintaining embodiment following the virtual hand metaphor.

In our design, we also built on previous work on remotely controlling cursors on TV screens. Pivot Plane-Casting defines a plane from the phone's orientation, anchored rigidly at the center of the virtual space [33, 34]. Rotating the phone rotates the plane and touching the screen allows attaching an object intersected by the plane before then translating it. Free Plane-Casting also casts a plane, however anchors the plane in a movable cursor position [33]. Touch input moves the cursor on that plane, thus translating the plane anchor itself and possibly an object along with it if attached. In Free Plane-Casting, phone rotations without concurrent touch have no effect on the cursor position as only the virtual plane orientation changes but not the cursor it is anchored in. INSPECT further extends plane-casting with a rotation mode that switches touch input from translating an attached object on the plane to rotating it [34]. Tiltcasting is related, but accounts for occluded objects [46]. By tilting the phone a plane shown on the screen is tilted correspondingly. Objects in front of the plane vanish and only objects intersected by the plane are selectable by using a cursor controllable by touch. HandyCast reuses the concept of a plane based on phone orientation introduced in Plane-Casting [33], but extends it in several regards: 1) In contrast to previous research, HandyCast is fully agnostic of the application state and does not require a dedicated integration for each VR application. This means, HandyCast works with any existing VR application that can be operated with two virtual hand avatars. 2) HandyCast transfers 10 dimensions (3D phone position, 3D phone orientation, and 2×2D touch), in contrast to 2D touch and 1D rotation as in Tiltcasting or

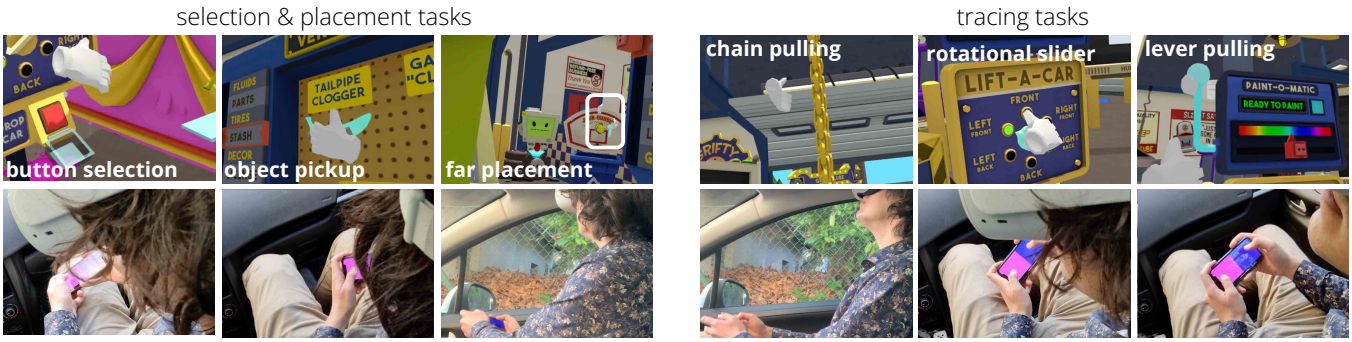

Figure 2: Using a single smartphone, HandyCast enables control over bimanual input tasks such as selection, placement, and tracing as found in common VR applications and games. Here we show representative tasks from Job Simulator [35].

3D rotation and 2D touch as in Pivot Plane-Casting and Free Plane-Casting. On the one hand, this means users can freely position and orient the plane and thus the virtual hands, following our hypothesis that this increases embodiment in VR. On the other hand, this enables bimanual selection and placement. In our comparison of HandyCast and a version of Tiltcasting adapted for bimanuality as one baseline, free plane placement with motion amplification led to faster task completion.

Pocket6 also uses the phone to select and manipulate 3D objects on a TV screen [7], using the phone to position a scaled 3D cursor and touch input to rotate an attached object (similar to INSPECT [34]). The ASP technique by Bergé et al. follows a similar approach while using external tracking [8]. HandyCast reuses the concept of position tracking as an input into our motion transfer for our hand avatar, but map multitouch input for further spatial hand control, whereas Pocket6 uses single touch for scene rotation and confirmation.

Taken together, HandyCast extends previous concepts, particularly a movable plane anchor as in Free Plane-Casting [33], but is first to use positional input, positional amplification, and up to two touch contacts to drive the 3D positions of two virtual hands in a 3D environment, co-located with the user in VR. This colocation with the user's body further requires spatial registration that depends on the user's seated position (rather than on the scene's or the cursor's center). Additionally, HandyCast targets low-effort input for control without looking at the smartphone itself.

3 POSE-AND-TOUCH TRANSFER FOR SMARTPHONE-BASED CONTROL

3.1 Problem and solution overview

HandyCast addresses the problem of space-efficient and bimanual object selection and manipulation in VR using a smartphone for input. Fundamentally, HandyCast derives 2×3D virtual hand positions and grab states from fusing the smartphone's 6D pose (i.e., 3D position and 3D orientation) with up to 2×2D touch locations when the phone is held like a gamepad (Figure 1b).

The mapping from phone pose+touch to two 3D hand positions in VR and their grab states is defined through our *pose-and-touch transfer function*, combining pose transfer, spatial touch transfer, and touch presence transfer. In pose transfer, we first compute a 3D plane, rotated according to the phone's rotation, and anchored in the amplified phone position. The amplified phone position is computed by non-linearly scaling the vector that leads from a calibrated neutral position to the current phone position. In spatial touch transfer, we then position two cursors on that plane based on the relative touch cursor input given to each touch zone on the phone's screen. Finally, we transfer touch presence for each touch zone, i.e., the binary state whether touch is applied or not, to the corresponding hand avatar's grab state. These three components can be computed and integrated at a single point in time, thus allowing simultaneous usage of phone and touch motion, individually for both hands.

3.2 Transfer functions to map phone input to two hand avatars in VR

In the following, we give further details on pose transfer, spatial touch transfer, and touch presence transfer.

3.2.1 Pose transfer. As shown in Figure 3, we constantly derive a plane in 3D space from the 6D smartphone pose, linking the real world to the virtual world in three steps.

1) Neutral pose anchoring. The user defines a neutral 3D phone position by pressing the volume-down button of the phone, e.g., when hands are recumbent in their lap. This calibrates a fixed spatial anchor $\mathbf{p}_{\text{anchor}}$ in world space (red sphere, 3b).

2) Phone tracking and phone vector computation. Moving the phone in the constrained space above the lap results in the phone vector $\mathbf{v}_{\text{phone}}$, leading from the calibrated anchor $\mathbf{p}_{\text{anchor}}$ to the phone's position. The physical movement is small (very short, white vector 3b). In HandyCast, phone motion can be tracked either through an outside-in (i.e., external) system, tracking a phone-mounted tracker with base stations, or through inside-out (i.e. standalone phone-only) tracking using the phone's sensors. To unify the phone's inside-out and the headset's coordinate system, we specify a registration procedure, as detailed in the supplementary material. By simply holding the phone in front of the headset and then pressing the volume-up button, a registration transform is created, used to convert from phone to headset coordinates.

3) Vector amplification allows the small phone vector to contribute to larger motions. We obtain the amplification vector $\mathbf{v}_{\text{plane}}$ by non-linearly scaling the phone vector $\mathbf{v}_{\text{phone}}$ as

$$\mathbf{v}_{\text{plane}} = \lambda \|\mathbf{v}_{\text{phone}}\| \, \mathbf{v}_{\text{phone}}.$$

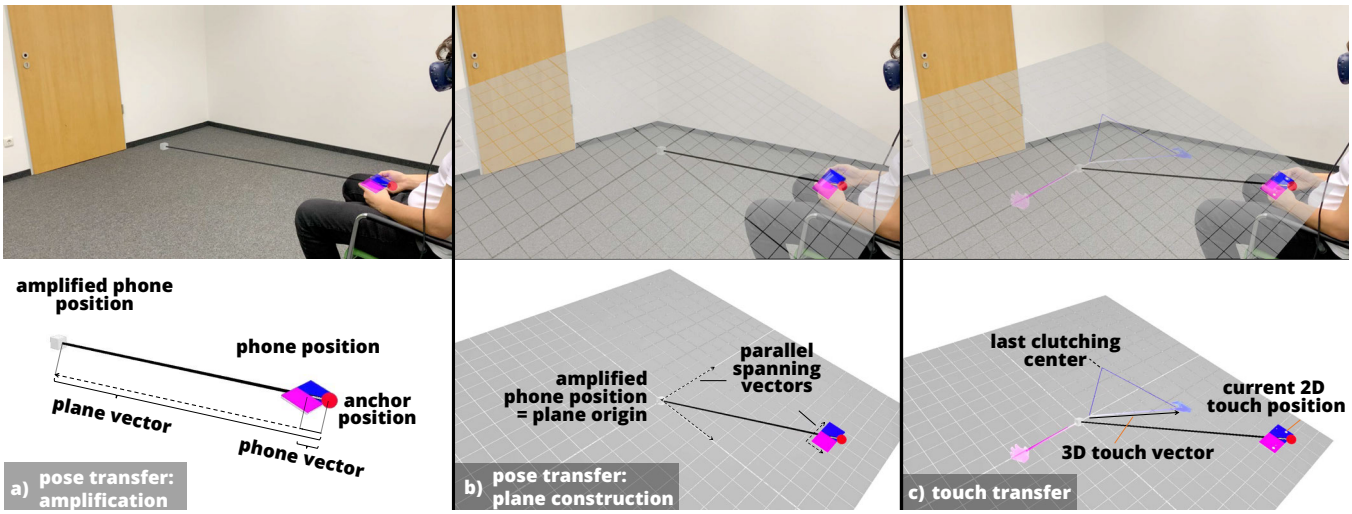

Figure 3: Schematic 3D representation of our *HandyCast* transfer function. For pose transfer, HandyCast spans a plane in 3D space, originating in the distance-amplified smartphone position. For touch transfer, the hands are moved from the plane origin (adjusted by a constant shoulder-width offset) along the gain-accelerated touch vector. Plane and vectors shown for illustration only.

Starting in the calibrated anchor $\mathbf{p}_{\text{anchor}}$ and following the amplification vector $\mathbf{v}_{\text{plane}}$ leads to the amplified phone position $\mathbf{p}_{\text{amplified}}$ (black vector in Figure 3b), serving as the plane's origin. λ is a constant, that can be set dependent on the desired strength of the amplification effect. In our studies, we choose $\lambda = 2.5$. From the phone orientation, we can easily obtain the phone's horizontal and vertical directional vectors (Figure 3c). Even without any touch applied, we offset the right hand position along the horizontal direction of the plane, and the left hand along the inverted horizontal direction, so to position both hands at shoulder width.

3.2.2 Spatial touch transfer. For spatial touch transfer, we divide the touchscreen into two separate control zones, allowing the thumb of either hand to independently control the respective hand avatar through touch transfer in five steps.

1) Initial touch transfer is zero. When no touch is present, virtual hand avatar position is determined through pose transfer only.

2) Gain-accelerated touch offset. We implemented a classical relative 2D cursor function computing the next 2D cursor offset from the current cursor offset and the current touch motion with gain acceleration following prior work [15, 40, 44]. Gain acceleration affords fine-tuning as well as longer-distance adjustments of hand avatars. Encroaching touch paths that cross the boundary between both touch zones, are maintained disruption-free for the thumb of the zone in which they originated. We implemented it as a finite state machine, instantiated once for each thumb, to compute each offset.

3) Transferring 2D touch offsets to 3D hand adjustments, we map the current 2D cursor offset to 3D touch vectors by rotating the offset vector according to the phone orientation.

Figure 3d shows an example of the resulting 3D touch vector.

4) Optional clutching supports continued touch transfer, which extends the virtual position, reachable by touch. When users need to move hand avatars beyond the constrained touchscreen input surface, they can lift off the thumb, and reposition it while the virtual hand stays in place. In our implementation of touch transfer, we allow for clutch timeout (set to 0.2s), i.e., a brief period of time within which a user may touch down again somewhere else on the screen and continue from the previous hand position. This updates the reference center for the next touch motion without affecting the already accumulated offset.

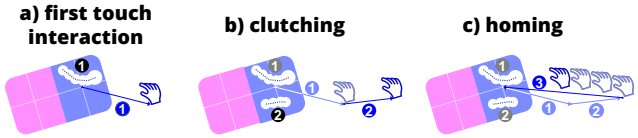

Figure 4: In 2D input space, we implement two gain-accelerated cursors that enable clutching and automatically perform homing.

5) Automatic homing finally compensates for potential clutching. To spare the user the manual effort of resetting hand avatars back to the initial position, the accumulated offset returns to 0 following a homing timeout (set to 2.5s). We avoid a sudden jump of hand avatars by lerping avatars ($t = 0.01, \Delta = 15$ms; ≈51% per second) back to their neutral position, defined by pose transfer only.

3.2.3 Touch presence transfer for commands. While VR controllers feature a trigger button to mimic grasping, smartphones offer no equivalent counterpart. Thus, we integrated grasp and release as part of touch contact itself—following the metaphor of direct touch interaction. If a touch-up event is not followed by another touch-down event in the respective control zone (i.e., either left or right) within the clutching delay, HandyCast triggers a release event.

3.3 Design decisions

3.3.1 Plane origin. HandyCast defines a plane in 3D space that is anchored in a position derived by amplifying the phone vector. This allows users to anchor the plane wherever optimal for the current bimanual task. In contrast, Tiltcasting [46] and Pivot Plane-Casting [33] anchor the plane origin in a fixed point of the 3D scene, so that the users can only touch and turn to move the cursor. Free Plane-Casting [33] makes no use of the phone position either and anchors the plane origin at the touch-controlled cursor unlike HandyCast's use of a point derived from the phone position.

TMMD [57] is the only technique where the user's body and the virtual scene are colocated, anchoring the plane origin with the phone. However, this still limits touch to object selection or requires physically walking up to the object, which is why TMMD supports raycasting. In contrast, HandyCast anchors the plane at a dynamic location computed through positional amplification, which can be controlled with little effort.

3.3.2 Pose transfer amplification. Go-Go uses isomorphic mapping when close to the body and non-linearly amplification beyond a threshold [47]. To maximize space efficiency, we design for immediate non-linear amplification, using touch transfer to allow for linear corrections at all times—even at distant locations.

3.3.3 Amplification anchoring. HandyCast bases pose transfer on real-world anchor positions. Using the head or a derived point (e.g., the chest) as a moving anchor similar to previous techniques leads to unintended hand avatar motions, particularly with large amplification. Thus, we initially create an anchor to world coordinates, defined as the user comfortably holds their phone in their lap.

3.3.4 Relative touch cursor mapping. HandyCast maintains a relative cursor for each touch zone where the touch-down location defines the origin of the touch interaction, rather than absolutely mapping the touch-down position to a coordinate. We chose this design as users have no visual control over their touch-down location on the screen when operating in VR.

3.3.5 Touch transfer acceleration. Touch transfer integrates gain acceleration for two reasons. 1) The constrained size of the screen requires touch motions to be small. Translating touch to medium and far distances would require high gains, but this contradicts our design goals to allow fine-tuning pose-transferred locations, which requires low gains. Gain acceleration offers both low and high gains, based on touch velocity. 2) Previous research has found gain acceleration advantageous over constant gain [15].

3.3.6 Clutching. Without clutching, lifting the thumb would reset the touch offset, thus instantly moving back the hand avatar to the pose-transferred location. Such liftoffs may even be unintentional, such as when sliding the thumb towards the screen edges.

3.3.7 Target-agnostic operation. HandyCast receives application-independent events as input, allowing our technique to operate on a standalone controller (i.e., phone), oblivious to the state of the virtual scene. That is, it operates without knowledge of target presence or proximity. While such information could improve the technique (e.g., whether the hand is hovering over an object), it would limit general applicability for real-world apps.

Figure 5: Users hold the HandyCast input phone like a gamepad, with each thumb operating in a single touch zone. For usage in common VR games, HandyCast can use inside-out tracking with no additional tracking hardware needed. For our two evaluations, a VIVE tracker is be mounted to the phone allowing us to separately investigate interaction performance and tracking effects.

4 IMPLEMENTATION

For the study setup, we implemented our transfer function in a VR evaluation environment, featuring objects, placement zones, and task protocols in Unity with the SteamVR framework and the Unity XR Interaction Toolkit. For input, we implemented an iOS 15 app. We smooth hand avatar positions using a 1€ filter [14]. The virtual environment supports user input from controllers and our phone app. The phone connects to the Unity server via TCP. In coordinates of the virtual environment, the phone pose can be determined from either the 6-DoF outside-in tracking by the VIVE system, or by the 6-DoF inside-out tracking provided by the phone only and also sent to the Unity server via TCP. The Unity app logs all transform updates of the HMD, the VIVE Tracker (Figure 5), the VIVE Pro Controllers, both hand avatars and all incoming pose and touch events from the smartphone. We also log all trial-related events with a variety of event parameters (e.g., positions, states, timestamps). The Unity app can also generate debug views, illustrating pose-and-touch transfer with planes, vectors, and ellipses in real-time, and shown throughout this paper. Separately and independently of our study environment, we developed a low-level controller driver for SteamVR, thus allowing backward-compatible drop-in use with standard VR apps on a Quest 2 with AirLink. Please refer to the supplementary material for a more detailed description of the controller implementation.

Inside-out tracking for mobile use. We integrated spatial tracking into HandyCast, such that our technique affords mobile use and needs not rely on an external tracking system. We build on ARKit [3] to track the phone inside-out, which performs visual-inertial odometry using the phone's rear-facing camera and the IMUs.

5 USER STUDY 1: ALL TECHNIQUES UNDER TECHNIQUE-OPTIMAL SPACE SETUP

The first study compared participants' performance in a series of target acquisition and placement tasks during unimanual and bimanual use in a within-subjects design. Our goal was to analyze the effect of technique on task completion time, physical space requirements, and trajectories. Participants also rated techniques on perceived exertion, perceived workload, and body ownership.

5.1 Input techniques

Participants completed all conditions using our *HandyCast* technique as well as two baseline techniques.[1] As a controller baseline, we included the state-of-the-art *Controller-based Go-Go* technique [47], where participants used *two* controllers and Go-Go's non-linear amplification for the tasks.

To maintain spatial comparability between subjects for the statistical analysis, we hold amplification constant across participants. Go-Go's amplification parameters D and k [47] specify the trade-off between 1) control volume, 2) accuracy, and 3) embodiment at distant locations. For this first user study, we set $D = 40$cm, and the amplification factor to $k = 0.5$, 1) ensuring that users can very comfortably reach all target selection and placement locations at short arm length, 2) with practical accuracy in the distance, 3) while maintaining the clear association between physical and virtual hand in an initially linear mapping, so to use Go-Go as a reference for embodiment across distant interaction. For *HandyCast* we specify $\lambda = 2.5$ (e.g., a forward motion of 15 cm is amplified to 5.56 m). Please refer to the accompanying video for a visual impression on the control volume.

Choosing a smartphone baseline requires more subtle consideration. The problem of two-hand input with a smartphone under space constraints in VR is unexplored. Thus, no readily applicable baseline for comparison with our novel technique exists. Instead, we have to adapt candidates from literature for bimanual usage, either based on a fixed-anchor technique such as Tiltcasting [46] and Pivot Plane-Casting [33] or a moveable-anchor technique such as Free Plane-Casting [33]. We chose a fixed-anchor technique over Free Plane-Casting for two reasons: 1) Pivot Plane-Casting as fixed-anchor technique is reported in the original paper to be faster than Free Plane-Casting. 2) Free Plane-Casting is ambiguous to adapt for two-hand input, given the need for two instead of one cursors, inducing interesting design questions. Should such a bimanual version of Free Plane-Casting anchor its freely moveable plane in a midpoint between both hands, making it most similar to our HandyCast technique? Or should this adaption maintain two fully independent planes? While any such design would be interesting to explore, it creates a novel method diffcult to consider an objective baseline.

In the space of fixed-anchor techniques, we choose Tiltcasting as a baseline, as we seek a baseline to provide fast and embodied input, both of which promised in the absolute mapping of Tiltcasting: 1) It promises fast input as touch-down positions define the target position directly reducing time needed for touch distances. 2) It promises embodied input as the neutral position is immediately assumed upon touch-up, not requiring thoughtful user input to go back to neutral.

Having chosen Tiltcasting, we extended it to *Bimanual Tiltcasting*, which anchors the plane fixed in space, and, therefore, uses a high control-display gain ratio for touch transfer. This allows reaching all objects in the scene. As described, we keep an absolute cursor for touch input. To enable bimanuality, we position both hand avatars on the same Tiltcasting plane with a small distance in between.

[1] In this study, we had included a second custom technique as a further condition, which turned out inferior to *HandyCast* and thus is not further reported on in this section. Note however that it was included in the statistical analysis to follow. Please refer to the supplementary materials for a technique description as well as the statistical analysis, also comparing it to HandyCast and the two baselines.

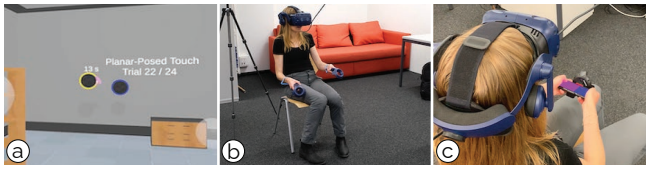

Figure 6: Apparatus for our full-range study. a) Our Unity app rendered the study environment, displayed through a VIVE Pro Eye. b) For *Go-Go* input, participants used the VIVE controllers. c) For all phone techniques, a VIVE tracker provided the pose of an iPhone 11 Pro, which relayed touches to Unity.

5.2 Apparatus: seated, input tracked outside-in

To remove the impact of tracking performance from this study, we used the HTC VIVE external tracking system for all techniques. (We separately evaluate the inside-out tracking of HandyCast and its effect on task completion in a tracking study.)

As shown in Figure 6, participants wore a VIVE Pro Eye for all tasks, sitting on a fixed chair. For *Controller-based Go-Go*, participants used both VIVE Pro controllers for input. For the smartphone-based *HandyCast* and *Bimanual Tiltcasting*, we mounted a VIVE tracker to an iPhone 11 Pro, which forwarded all touch events registered on the phone to a PC. Participants do not see the debug view (showing planes or vectors) before or during the study but only see the hand avatars as such when training and using our technique.

5.3 Task

Participants completed two tasks in the study. For both, they were instructed to complete them as fast as possible.

Task 1: unimanual object selection and placement. During each trial, participants were instructed to grab a highlighted object in the virtual scene with the specified hand. After grabbing it, they moved it to the indicated placement zone. Releasing the object within the zone completed the trial and advanced to the next. When participants erroneously dropped the object, they could grab and move it again until they succeeded.

To acquire a target, participants either pressed the trigger on the controller or touched and held down on the smartphone. Releasing the trigger or the touch dropped the virtual object.

An object counted as correctly placed if its center was within the placement zone, moved with the instructed hand. Participants received visual cues as soon as a release would be correct (Figure 7c). An acoustic cue then confirmed a successful release.

Task 2: bimanual selection and placement. Participants grabbed two targets during each trial, one after the other, and then placed them in the corresponding placement zones. Consistent coloring indicated which target to grab with which hand (left: pink, right: blue) and where to place it. Our video figure shows examples of bimanual trials, each of which followed one of the following combinations: $\{\text{grab}_L, \text{grab}_R, \text{drop}_L, \text{drop}_R\}$, or $\{\text{g}_R, \text{g}_L, \text{d}_R, \text{d}_L\}$.

Figure 7: Participants completed unimanual and bimanual tasks, a) selecting from a $H3 \times W5 \times D3$ grid and placing at a different grid location (every second target and placement shown for clarity.) Targets and placement zones were equally distributed across trials. b) Acquisition during Task 1 (unimanual), c) first placement of Task 2 (bimanual). The left hand is colored in magenta, the right in blue. The target and placement zones are color-coded correspondingly when highlighted.

In both tasks, for each trial, participants had 15 seconds to acquire a target and 15 seconds to place it. The remaining time was indicated by a countdown timer. If participants ran out of time, the trial counted as an error.

Target arrangement. Figure 7 shows the grid arrangement of locations for targets, which were shown one and two at a time for Task 1 and 2, respectively. Placement zones were at the same grid locations, one or two highlighted depending on the task.

The location of targets built on previous studies (e.g., Erg-O [43] and other VR input techniques [19, 56]), but added placement zones, distant object interaction, and bimanual interaction. In this study, the target size is set to 25 cm. The final grid (height = 3 × width = 5 × depth = 3) contained cells of 1.6 m ×1.6 m ×3 m with a 0.4 m spacing in 2D and 1 m spaces in depth.

Technique rating. After completing all trials for a technique, participants filled out a short questionnaire in VR: 2 Borg CR-10 ratings (exertion in hands/lower arms, exertion in upper arms) from 0 (nothing at all) to 10 (extremely strong), 4 TLX subscales (mental demand, physical demand, effort, frustration) from 1 (very low) to 20 (very high), and 4 avatar embodiment questions (ownership, double hand, control, interference) [20] on a 7-point Likert scale from 1 (strongly disagree) to 7 (strongly agree).

5.4 Procedure and design

Before the evaluation phase, participants received training with all techniques in both tasks for approx. 10 minutes. After finishing a technique in a task, they answered the questionnaire. Before starting the measured study trials for the next technique, they were allowed at least two more training trials to remember it. The evaluation part took approximately 40 min per participant.

Independent variables. The study followed a within-subjects design with two independent variables: Operation and Technique. Operation had two levels: *Acquisition* and *Placement*. Technique is considered with three levels: *HandyCast*, *Controller-based Go-Go*, *Bimanual Tiltcasting*. Technique order was counterbalanced across participants and both tasks with a Latin square.

For each task, we precomputed randomly selected target and placement locations, which were counterbalanced across participants. Even though we did not explicitly analyze them as independent variables, we ensured equal distribution of Hand (Task 1, *left* or *right*) or Hand Combination (Task 2, start with grab_L or grab_R, see above), and Depth delta with three levels (0: same layer, +1: pushing back, −1: bringing forward).

Trial repetitions. For Task 1, participants repeated acquisition and placement 7 × for 2 Hand × 3 Depth delta × 4 Techniques × 2 Operations = 336 trials per participant. For Task 2, they repeated the task 4 × for 2 Hand × 3 Depth delta × 4 Techniques × 2 bimanual Operations = 192 trials per participant.

Dependent variables. To analyze differences between techniques and operations, we logged task completion times (i.e., selection and placement time), as well as physical and virtual motion paths. For *Controller-based Go-Go* in Task 2, we measured the sum of both controllers' motions; for the smartphone techniques, we doubled the length of paths traveled to account for movement of both real hands holding the smartphone. From the recorded motion paths, we derived the required volume for operation (i.e., control space) from the enclosing world-oriented cuboid, based on the 5th to 95th percentile of points per axis.

5.5 Participants

We recruited 12 participants (2 female, 10 male, ages=23–35, M=28.4, SD=4.1). 4 participants had never worn a VR headset before, 5 used one less than 5 times, 2 occasionally, and 1 on a weekly basis.

5.6 Results

5.6.1 Completion time. We performed a two-way repeated-measures ANOVA on task completion time for Technique × Operation with participant as the random variable.

In Task 1, participants completed acquisition on average in 2.49 *s* ($\sigma = 0.72$) and placement in 2.74 *s* ($\sigma = 0.73$). We found a significant main effect of Technique on time ($F_{3,33} = 27.331, p < .001, \eta^2 = .713$) as well as of Operation on time ($F_{1,11} = 30.592, p < .001, \eta^2 = .736$). We also found an interaction between both variables ($F_{3,33} = 11.472, p < .001, \eta^2 = .510$). Post-hoc t-tests using Bonferroni-adjusted confidence intervals on Technique showed

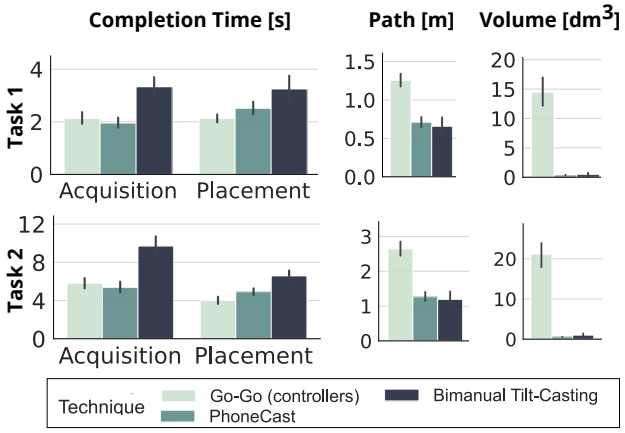

Figure 8: Aggregated results of our study by task and operation, and the dependent variables completion time, length of input paths, and physical control space volume. Error bars indicate 95% confidence intervals.

significant differences between *Controller-based Go-Go* and *Bimanual Tiltcasting* as well as between *HandyCast* and *Bimanual Tiltcasting*, but not between *Controller-based Go-Go* and *HandyCast*. As shown in Figure 8, *Controller-based Go-Go* was 1.2 s faster than *Bimanual Tiltcasting* on average (154.8%, $p < .001$), whereas *HandyCast* was 1.1 s faster than *Bimanual Tiltcasting* (147.4%, $p < .001$). We found no significant difference between *Controller-based Go-Go* and *HandyCast*.

In Task 2, participants completed acquisition (accumulated for both objects) on average in 6.91 s ($\sigma = 2.16$) and placement in 5.43 s ($\sigma = 1.35$). We found a significant main effect of Technique on time ($F_{3,33} = 60.028, p < .001, \eta^2 = .845$) and of Operation on time ($F_{1,11} = 111.130, p < .001, \eta^2 = .910$). We also found an interaction effect between both ($F_{3,33} = 38.793, p < .001, \eta^2 = .779$). Post-hoc t-tests using Bonferroni-adjusted confidence intervals showed a significant difference between all comparisons involving *Bimanual Tiltcasting*. We found no significant difference between *Controller-based Go-Go* and *HandyCast*.

5.6.2 Travel length of physical motions. For Task 1, a one-way repeated-measures ANOVA on traveled motion path for Technique showed a significant main effect ($F_{3,33} = 79.053, p < .001, \eta^2 = .878$). Post-hoc t-tests using Bonferroni-adjusted confidence intervals do reveal significant differences between *Controller-based Go-Go* and both of *HandyCast* and *Bimanual Tiltcasting*, but not between *HandyCast* and *Bimanual Tiltcasting*. As shown in Figure 8b, the average distance traveled using *HandyCast* was 0.544 m shorter than *Controller-based Go-Go* (56.6%, $p < .001$). Using *Bimanual Tiltcasting*, the average distance was 0.597 m shorter than *Controller-based Go-Go* (52.4%, $p < .001$).

For Task 2, we also found a significant main effect ($F_{3,33} = 104.622, p < .001, \eta^2 = .905$). Post-hoc t-tests using Bonferroni-adjusted confidence intervals showed significant differences in the same comparisons as in Task 1.

5.6.3 Required physical volume (control space). For Task 1, a one-way repeated-measures ANOVA for Technique on the required volume found a significant main effect ($F_{3,33} = 97.763, p < .001, \eta^2 = .895$). As shown in Figure 8c, post-hoc t-tests using Bonferroni-adjusted confidence intervals revealed significant differences between all techniques except between *HandyCast* and *Bimanual Tiltcasting*. *HandyCast* required 14014.1 cm^3 less control volume than *Controller-based Go-Go* (2.72%, $p < .001$).

For Task 2, we also found a significant main effect ($F_{3,33} = 123.414, p < .001, \eta^2 = .918$). Post-hoc t-tests using Bonferroni-adjusted confidence intervals showed significant differences between the same combination of techniques as in Task 1. *HandyCast* required 20547.9 cm^3 less control volume than *Controller-based Go-Go* (2.79%, $p < .001$).

5.6.4 Questionnaires. Next, we analyze the participants' questionnaire responses. Please refer to the supplementary material for a detailed break-down of answers for all questions.

For perceived exertion (Borg CR-10), pairwise Bonferroni-adjusted Wilcoxon signed-rank tests showed no significant differences between participants' ratings ($p < .1$).

In terms of perceived workload (NASA TLX), pairwise Bonferroni-adjusted Wilcoxon signed-rank tests reveal a significant difference in *effort* between *Controller-based Go-Go* and *Bimanual Tiltcasting* (7.25 ± 4.0). On average, effort in *Controller-based Go-Go* was perceived lower by 3.75 than in *Bimanual Tiltcasting* ($p = .033$).

For avatar embodiment, pairwise Bonferroni-adjusted Wilcoxon signed-rank tests showed significant differences between *Controller-based Go-Go* and *Bimanual Tiltcasting* for both *Ownership* and *Control*. Reported *Ownership* was on average 2.5 points higher in *Controller-based Go-Go* ($p = .029$), and reported *Control* was on average 2.3 points higher in *Controller-based Go-Go* ($p = .031$).

5.6.5 Error rates. Participants rarely ran out of time: 3 times of all 4032 trials conducted in Task 1 (once with *Controller-based Go-Go*, twice with *Bimanual Tiltcasting*), and 6 times in Task 2 (among these: once with *Controller-based Go-Go*, and 4 times with *Bimanual Tiltcasting*).

5.7 Discussion

Our first evaluation revealed several interesting insights about participants' performance during interaction with objects in VR using the different techniques.

Insight 1a: Controller-based Go-Go and HandyCast achieved comparable completion times, but HandyCast requires less motion. Performance of HandyCast was closest to the well-established Go-Go with no significant difference between them. However, despite the comparable completion times, HandyCast required significantly less travel on average, both for unimanual (0.54 m, 56.6% of Go-Go) and bimanual tasks (1.37 m, 48% of Go-Go). The volume enclosed by HandyCast was equivalent to a cube with edge length 8.4 cm. Go-Go's volume is equivalent to a cube with edge length 28 cm (> 3×). These results are particularly interesting, since the indicated path lengths already represent the phone trajectory multiplied by two to account for the movements of both hands, holding the phone. These results support our design intentions, in particular for HandyCast's transfer function.

Insight 1b: participants do not report significant differences between Controller-based Go-Go and HandyCast. Participants' questionnaire responses are similarly promising, as HandyCast showed no significant loss in *ownership* or *control* in the embodiment questionnaire ratings, compared to the individual controllers. At the same time, despite the reduction in control space, participants' ratings also showed no difference in perceived fatigue or physical demand between Go-Go and HandyCast. It was one of our hypotheses that HandyCast would reduce fatigue. Taken together, this indicates that ownership, control, and demand in our HandyCast technique are similar to input with two controllers in current VR systems, yet with the benefit of ubiquitous use, in mobile settings, and at a fraction of the required space for reliable operation.

Insight 1c: Touch presence transfer for commands makes placement slower than selection. As described, earlier, we found an interaction effect between Technique and Operation on *completion time*. For Task 1, i.e., a unimanual operation of first selecting a single target and the placing it, *Controller-based Go-Go* showed no difference between acquisition and placement completion time (both $M = 2.1$ s), suggesting that there is no structural difference between the two operations. However, in *HandyCast*, we observe a difference of 0.5 s between selection ($M = 2.0$s) and placement ($M = 2.5$s). We rationalize this with two effects. First, HandyCast is more susceptible to "loosing" objects "on the way", entailing a reacquisition. While a trigger button in the *Controller-based Go-Go* controllers can be hold tight when moving the virtual hand with the object attached from the selection location to the target location, HandyCast employs touch for both attaching to the object and then also to move it, together with pose transfer. When users need to clutch on the touchscreen, the objects remains attached for 0.2 s before being released. If clutching in motor space takes longer and the user exceeds this timeout, the object is dropped, necessitating a reacquisition, and thus slowing down overall completion time. Second, when participants had not sufficiently internalized the 0.2 s delay, they pulled the phone back too quickly (with the object still attached), thus requiring a re-acquisition and another placement attempt for the object.

Insight 1d: Independent Go-Go controllers allow concurrent movement and thus slightly faster placement. Comparing the results of Task 1—in particular the interaction effects—with Task 2 also reveals interesting aspects. Bimanual acquisition is slower than double the unimanual acquisition time, likely because 1) users need to connect the color coded outline to the color-coded hand avatar, inducing cognitive processing time, and 2) users need to coordinate which hand to move to the targets, and how to position the other hand in the mean time. *HandyCast* was faster than *Controller-based Go-Go* for acquisition (−0.4 s), but slower for placement (+0.9 s).

Beyond the reason above, the motion logs showed another reason for Go-Go's advantage. Participants often acquired targets sequentially, but often *concurrently* moved both hands to the target zones. While *HandyCast* provides the same effect during pose transfer, touch transfer is slowed down when the pose transfer moves in an opposite direction.

Insight 1e: Bimanual Tiltcasting requires the least volume and the least motion paths of all techniques, but is slower having only touch and orientation input available. Given that all differences between smartphone techniques were significant with respect to completion time, we can establish a ranking: *HandyCast* was fastest, followed by *Bimanual Tiltcasting*, in both Tasks 1 and 2. The mean acquisition time of approx. 3.4 s, which we measured in Task 1 under the *Bimanual Tiltcasting* technique is similar to the acquisition measured at 3.6 s in the original Tiltcasting paper under the target-agnostic, standard-display, small-target condition ([46], Table 1). While the differences in the concrete task and technique (we evaluated our *bimanual extension* of the original technique, see Subsection 5.1) do not allow for further comparison, the equivalent magnitude in completion time is an indication of external validity. The fact that completion under the bimanual task is more than twice the unimanual time might follow the same considerations described in *Insight 1d*.

In terms of control volume and motion paths, *Bimanual Tiltcasting* was the best-performing technique, making it a suitable technique for interaction in space-constrained settings. Yet, without a significant increase in control volume, *HandyCast* was significantly faster (1.1 s in Task 1 and 2.99 s in Task 2). This indicates that *HandyCast* manages to leverage our assumption that users *inadvertently* use body motions, even when the controller is not motion-sensitive.

Bimanual Tiltcasting differs from *HandyCast* in two aspects, which might contribute to *Bimanual Tiltcasting*'s slower task completion time. First, users can only use one channel–touch–to move hands forward in the former, while they can use two channels–touch and pose–in HandyCast. This, effectively reduces the bandwidth of information the user can input at a given point in time. Second, the absolute cursor, designed in Tiltcasting is less effective than our relative and gain-accelerated spatial touch transfer: touch-down events carry uncertainty where the hand will actually jump to, and accidental touch-up events reset the hand to the world-anchored neutral position, requiring to reacquire the object starting from the neutral position again.

In summary, either changing touch control or adding pose control or changing both made the difference. Thus, the question might arise if changing touch control only would have provided sufficient improvement of *Bimanual Tiltcasting*, rendering the addition of pose control superfluous. To understand the relative important between the input modes, we consider Figure 8, Task 2, which reveals that the physical motion path span approx. 60 cm with *HandyCast*. Given that a physical forward motion of 15 cm amplies to approx. 5.5m of forward motion in virtual space in studyplanecast, we can conclude that pose input was the driving input mode to our technique in order to cover the long-ranging virtual distances.

Taken together, using touch and orientation only without our full 6DoF pose transfer to move the hands was detrimental to the performance. We interpret this ranking as a promising indicator of our pose-and-touch transfer function.

Summary

Taken together, *HandyCast* yields completion times comparable with the best baseline *Controller-based Go-Go*, but significantly reduces motion and control space. Yet, we do not observe a loss in reported ownership, control, demand, or effort, between the two

techniques. Compared to using touch and orientation only, our concept of 10D pose-and-touch transfer enables significantly faster completion times with only an insignificant increase in control volume.

6 USER STUDY 2: SPACE-CONSTRAINED SETUP

In our first study, we evaluated user performance for the controller-based Go-Go baseline, a smartphone baseline, and *HandyCast* proposed in this paper. In accordance with the original idea of Go-Go, the Go-Go amplification parameters were configured such that the farthest target in the scene could be reached by an arm length. However, in our envisioned scenario of mobile VR usage, space might be significantly smaller. Therefore, we conduct a second study in a space-constrained setup, comparing a more sensitive configuration of Go-Go (D = 35cm, k = 4) with earlier and stronger amplification against *HandyCast* (same parameters as in study 1).

6.1 Procedure, and Apparatus: seated, space-constrained input tracked outside-in

For this second study, we physically constrain the space with cardboard to the sides and a wall to the front of the seated user, mimicking the available space on a bus or airplane seat as shown in Figure 9. Again, we use the outside-in VIVE tracking system for both techniques, making sure that the base stations can see into the boxed setup by mounting them on the ceiling.

Figure 9: Space-constrained setup for the second user study.

6.2 Task, Procedure, and Participants

In this study, 20 participants (3 female, 17 male, ages=21-41, M=29.0, SD=6.3), complete Task 2 from our first study (bimanual acquisition and placement). 6 of the participants never used VR, 10 use it a few times a quarter or less, and 4 weekly. We follow the same procedure as in the first user study.

6.3 Design

This study follows a mixed design, with TARGET SIZE as a between-subject variable, and TECHNIQUE as a within-subjects variable. TARGET SIZE has two subject groups: *large* (25 cm) and *small* (20 cm) with 10 subjects each. TECHNIQUE has two levels: *Space-Constrained Go-Go* and *HandyCast*. We're interested in the same dependent variables as in user study 1, i. e., *completion time*, *interaction volume*, and *motion path length*. We counterbalance the order of TECHNIQUE by alternation.

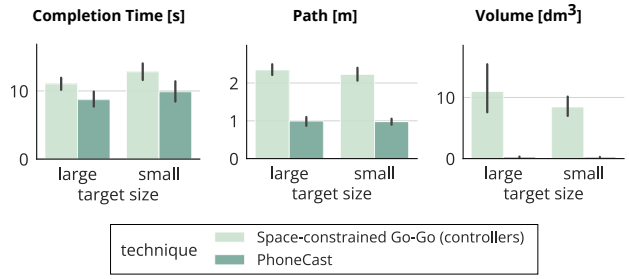

Figure 10: Aggregated results of our space-constrained, seated study by target size. Error bars indicate 95% confidence intervals.

6.4 Results

We performed a mixed-design ANOVA on task completion time with TECHNIQUE as independent within-subjects variable and TARGET SIZE as independent between-subject variable. Figure 10 gives an overview.

We found a significant main effect of TECHNIQUE on time ($F_{1,18}$ = 36.9, $p < .00001$, $\eta^2 = .67$) as well as of TARGET SIZE on time ($F_{1,18}$ = 3.3, $p < .1$, $\eta^2 = .15$). Post-hoc t-tests using Bonferroni-adjusted confidence intervals showed a significant difference between large and small targets when using Space-Constrained Go-Go ($p < .1$) but not when using HandyCast.

With respect to motion, we found significant main effects of TECHNIQUE on path length ($F_{1,18} = 453.4$, $p < .00001$, $\eta^2 = .962$) as well as on volume ($F_{1,18} = 71.9$, $p < .0001$, $\eta^2 = .8$)

6.5 Discussion

This second study reveals several insights, in particular against the backdrop of the findings of study 1.

Insight 2a: In a space-constrained setup, HandyCast enables faster object selection and placement than Space-Constrained Go-Go. While there was no significant difference between HandyCast and unconstrained Go-Go in the previous study, in the space-constrained setup of this second study, HandyCast (M = 9.2s) is significantly faster than Space-Constrained Go-Go (M = 11.8s) by 22.0%. This drop in performance of the Go-Go technique results from the loss of control due to increased amplification compared to constrained Go-Go. While this increase in amplification is required to stay within the space constraints, it entails at least two effects detrimental to Go-Go's performance: 1) While Go-Go's amplification function is smooth and continuous by its mathematical formulation, the more aggressive amplification accelerates the virtual hand much quicker, making it harder for users to supervise ballistic movements, and thus increasing the risk of overshooting. In HandyCast, the same or even a smaller volume can be operated at lower motion amplification because input is complemented by touch. 2) At distance, the Go-Go technique operates at high amplification which then also impedes the corrective movement after the initial ballistic movement. In HandyCast, corrective movements with touch operate at constant speed, independent of distance, therefore always guaranteeing the ability of fine-tuning.

Insight 2b: Even with its aggressive amplification, Space-Constrained Go-Go uses significantly more space than HandyCast. By design, Space-Constrained Go-Go requires less interaction volume and shorter path lengths than standard Go-Go in the previous study. Interestingly, despite not changing the parameterization of HandyCast, users also travel less with the smartphone and occupy less interaction volume than in the previous study, likely due to their awareness of being constrained and thus intuitively relying on touch input more to avoid touching the obstacles. However, HandyCast still requires significantly less interaction space ($M = 0.3dm^3$) and path lengths ($M = 0.99m$) than Space-Constrained Go-Go ($M = 9.95dm^3$ and $M = 2.3m$ resp.). From this, we conclude that HandyCast exhibits a fundamental space-efficiency advantage, independent of the specific Go-Go parameterization. This advantage results partly from HandyCast's ability to accept touch input, but also from the specific designs concerning amplification reference points: While Go-Go computes *two amplification vectors* relative to *a single point* at the chest, HandyCast computes *a single amplification vector* relative to *a single point*, namely the custom-defined home position in a neutral posture.

Because both Go-Go controllers share the same reference point, Go-Go introduces a radius of linear amplification in front of the chest which smoothly transitions to a non-linear amplification. If non-linear amplification was to kick-in immediately, only one controller could be set to a neutral position at the reference point, blocking the other controller from the neutral position, forcing it to operate at non-linear amplification. Using a single smartphone allows for a natural neutral position from which non-linear amplification can kick-in immediately while still maintaining bimanuality through touch.

Insight 2c: Decreasing the interaction volume in Go-Go induces more frequent controller collsions. In contrast to the previous study, in this space-constrained setup, multiple participants (P3, P4, P8, P19) reported that it was annoying or irritating that they often hit one controller with the other when selecting or placing objects. This effectively hints at the reduced bimanuality when using controllers in a shared small volume, esp. when oriented around a shared reference point. By design, a single smartphone as in HandyCast cannot suffer from this problem.

Insight 2d: Space-Constrained Go-Go benefits from larger target sizes due to larger amplification inaccuracies at distance. As indicated by the interactions reported above, Space-Constrained Go-Go benefits from larger targets with respect to completion times whereas HandyCast does not. We interpret these results as follows: A trial is composed of 1) pointing towards the target 2) attaching to it, 3) pointing towards the drop zone with the attached target and 4) detaching from the target for each corresponding hand. Pointing in turn is composed of initial and final movement. The target size only impacts the final movement in the pointing steps. Thus, the fact that Space-Constrained Go-Go's performance is significantly impaired by reducing target size reveals that the final movement step is impaired. The fact that HandyCast is not equally impaired indicates that the final movement is less impacted here, either due to the ability to fine-tune virtual hand positions quickly by touch input, or due to the lower amplification, afforded by the two input modalities that can be combined to high amplifications if and only if desired.

Summary

While there was no significant difference between HandyCast and standard Go-Go in the previous study with respect to completion time, this second study has revealed that HandyCast outperforms Space-Constrained Go-Go in a spatially constrained setup by unfolding its advantage of decomposing non-linear and linear amplification into two different, separately controllable modalities of hand motion and thumb touch input. It's fundamental property of simulating bimanuality through touch-augmented pose transfer becomes an advantage over Space-Constrained Go-Go which suffers from controller collisions reducing independent input in a constrained space.

7 TRACKING STUDY: PHONE VS. VIVE

In our previous studies, we evaluated participants' performance using the input techniques under ideal tracking conditions. The purpose of this evaluation was to assess the effect of tracking technology on performance. HandyCast has the potential to run entirely on the user's phone, where inside-out tracking may cause less accuracy in task completion and thus reduced completion speed.

Task, procedure, and apparatus. In this study, participants completed Task 2 from our first study (bimanual target acquisition and placement). Using *HandyCast*, we compared two tracking methods: outside-in tracking and inside-out tracking. Outside-in tracking used the VIVE's surrounding base stations and a tracker attached to the phone as in our first study (Figure 5). Inside-out tracking used our ARKit-based implementation.

The evaluation implemented a within-subjects design with Tracking Method as the independent variable. Participants repeated bimanual acquisitions and placements 24 × (2 Tracking Method × 2 blocks) = 96 trials. The order of Tracking Method was counterbalanced across both tasks and participants. Participants received the same instructions and training as in our first study, with the addition not to accidentally occlude the phone's back camera.

The apparatus was the same as in our first study, only that inside-out tracking as well as outside-in tracking were active throughout all trials for later comparison. In the *Outside-in* condition, the VIVE tracker drove the transfer function as in our first study. In the *Inside-out* condition, the tracker merely served to record ground-truth positions and orientations, but the transfer function was exclusively driven by the phone-reported 6D poses without input from the VIVE system. Participants performed this spatial registration before each of the two *Inside-out* blocks.

Our analysis is two-fold. 1) We investigate the total drift across a full block of 24 trials with inside-out tracking. 2) We analyze the effect of Tracking on Completion Time as a dependent variable.

Participants. We recruited another 4 participants (1 female, 3 male, ages=27–32, M=28.8, SD=2.2). 1 participant had used VR occasionally, 2 less than 5 times, and 1 never.

Results—Tracking effect on performance. We performed a one-way repeated-measures ANOVA on *completion time* for Tracking Method. Figure 11 shows the distribution of completion times, showing a

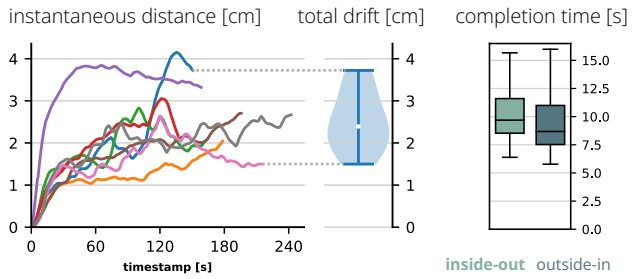

Figure 11: Tracking evaluation. Left: Drift that emerged from phone-based *Inside-out* tracking during a block. Right: Impact of tracking method on task completion (p = .1).

lower mean for *Outside-in* (M=9.85 vs. *inside-out* M=10.49), though differences were not significant.

Results—Spatial drift. As shown in Figure 11a, using *Inside-out tracking* incurred an average total drift of 25 mm (SD = 7.3) during a full block. With an amplification factor λ = 2.5, a difference of 25mm creates an error offset of approx. 12 cm in virtual space, when reaching out exactly 10 cm. For each *Inside-out* block, we computed the Euclidean distance between the VIVE tracker and the phone-reported position, offset to zero upon the start of the first trial. Figure 11 left shows a plot of all 8 blocks. For visual clarity only, in the plot, we also apply exponential smoothing with a factor of .001. The distribution of accumulated drift at the end of each block is shown in Figure 11 middle. As measured by the tracker, the inside-out-tracked pose drifted 0.25 cm/m on average (SD = 0.09), computed as accumulated error divided by total distance traveled.

Discussion. Our analysis showed the impact of drift as part of *Insight-out* tracking on the performance of using *HandyCast*. Over the course of just one block, phone-reported positions drifted up to 4 cm. The only small difference in completion time indicates that participants were able to compensate for this amount of drift, possibly because of the dominance of vision over proprioception during interaction in VR.

Depending on a participant's speed, a block lasted 2.5–4 minutes. A gameplay session in VR may last much longer and more drift may accumulate as a result. HandyCast could compensate for such drift by adjusting the respective anchor position to calculate pose transfer. This is currently static and future iterations of HandyCast will need to account for dynamically updated anchors (e.g., by monitoring maximum proximity to the torso or detecting resting arms in the lap), which would also allow resetting drift.

Some participants' drift curves show sudden moments (e.g., violet or blue trace). During the trial, these occasional moments manifested as visual jumps and thus interfered with the participant's current input motion. This demanded participants' manual counteraction, which slowed down the trial in these specific moments.

Overall, our evaluation has established the feasibility of phone-based inside-out tracking for the use of HandyCast. This also makes the use of our phone-only controller driver practical, which allows control over existing VR apps and games. We also believe that with increasingly powerful tracking methods on today's consumer phones, drift will decline in the future.

8 LIMITATIONS AND FUTURE WORK

Improving Tracking. In this paper, we have proposed full inside-out tracking based on the phone's sensors for pose-and-touch transfer and compared it to outside-in tracking for reference. To improve the inside-out tracking accuracy, future research lies in using a headset-based tracking, e.g. by camera-based phone pose estimation, leveraging hand tracking as a proxy to then estimate the phone pose from wrist poses, by displaying an optical marker to the blindly operated touchscreen, or even using acoustic tracking [29, 54].

Studying Transfer Functions under Different Parameters. Since HandyCast is scene-agnostic, all parameters can be specified on the controller side, similar to setting the gain acceleration of a mouse in the operating system. For specific VR apps, users might choose a different amplification factor in HandyCast, depending on the virtual and physical environment. In this paper, we have specified values through pilots so that users can comfortably reach all targets in the task with all pose-aware techniques. Future research lies in understanding the effects of different parameters within our design, e.g., the relative importance between pose and touch by studying it under varying gain factors, amplification factors or even other amplification function families (e.g., Hermite curves, [56]) in the case of pose transfer. Furthermore, exploring the coupling of positional or positionally amplified input with other plane-anchoring techniques such as Free Plane-Casting [33], e.g. maintaing one plane per cursor, offers promising research directions.

Designing for More Complex Interactions. Using the 10 DoF from pose and touch in a single smartphone constrains HandyCast to positional bimanuality, not sufficient for full 12-DoF bimanuality, e.g., required to hold a bottle in one hand and opening it with the other. Other more complex interactions, however, could be enabled by processing further input signals. In particular, we use atomic touch-down and touch-up events to attach to and detach from objects, e.g., leaving double taps unused for further mappings. In the future, double taps might be used to enter the teleportation mode, thus enabling rested locomotion.

9 CONCLUSION

We have presented HandyCast, a smartphone-based input technique designed to control bimanual input in large VR environments through small motions in space-constrained environments. HandyCast's core concept is its pose-and-touch transfer function that fuses the smartphone's 3D position, 3D orientation, and 2D touch motions as input to jointly output two independent 3D positions to place virtual hand avatars. The touch-transfer component of our technique allows users to clutch during input, thereby repeatedly readjusting avatar locations, and thus affording navigation inside infinite spaces.

In our *space-constrained user study*, we found that HandyCast requires significantly less completion time, path length and interaction volume compared to the space-constrained Go-Go implementation. In a *tracking evaluation*, we compared the performance of

HandyCast during externally tracked and phone-only tracked operation. We found that despite small inaccuracies and drift, HandyCast affords operation on today's smartphones and therefore enable fully mobile use together with untethered VR systems.

Because HandyCast is completely scene-agnostic, our low-level SteamVR driver is fully compatible with existing VR applications and games, which we demonstrated at the example of Job Simulator. We conclude that HandyCast brings comfortable, full-range, and bimanual 3D input to mobile VR by retrofitting the user's smartphone as an ubiquitous controller. In comparison to controller-based state-of-the art baselines, HandyCast does not only reduce interaction volume and improve completion times in space-constraint settings, but 1) does not require dedicated hardware, 2) can be highly amplified for usage in very small volumes without suffering from inter-controller collisions, 3) allows unlimited reach without parameter re-adjustment through touch, and 4) allows for robust refinement of virtual hand position through touch, independent of the virtual distance between user and object.

ACKNOWLEDGMENTS

We thank Reinhard Schütte, Tobias Grosse-Puppendahl, and Jochen Gross for their support. We also thank Danil Ivanov for early explorations of phone-based interaction in Virtual Reality with us.

REFERENCES

[1] 2019. National Geographic Explore VR on Oculus Quest. https://www.oculus.com/experiences/quest/2046607608728563/
[2] 2021. Virtual Reality Movie Studio. https://www.felixandpaul.com/
[3] Apple. 2022. ARKit. https://developer.apple.com/augmented-reality/arkit/
[4] Ferran Argelaguet and Carlos Andujar. 2013. A survey of 3D object selection techniques for virtual environments. *Computers & Graphics* 37 (2013), 121 – 136. https://doi.org/10.1016/j.cag.2012.12.003
[5] Rahul Arora, Rubaiat Habib Kazi, Fraser Anderson, Tovi Grossman, Karan Singh, and George W Fitzmaurice. 2017. Experimental Evaluation of Sketching on Surfaces in VR.. In *CHI*, Vol. 17. 5643–5654.
[6] Mahdi Azmandian, Mark Hancock, Hrvoje Benko, Eyal Ofek, and Andrew D Wilson. 2016. Haptic Retargeting: Dynamic Repurposing of Passive Haptics for Enhanced Virtual Reality Experiences. *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 1968–1979. https://doi.org/10.1145/2858036.2858226
[7] Teo Babic, Harald Reiterer, and Michael Haller. 2018. Pocket6: A 6DoF Controller Based On A Simple Smartphone Application. *Proceedings of the Symposium on Spatial User Interaction*, 2–10. https://doi.org/10.1145/3267782.3267785
[8] Louis-Pierre Bergé, Emmanuel Dubois, and Mathieu Raynal. 2015. Design and Evaluation of an" Around the SmartPhone" Technique for 3D Manipulations on Distant Display. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction*. 69–78.
[9] Joanna Bergström, Tor-Salve Dalsgaard, Jason Alexander, and Kasper Hornbæk. 2021. How to Evaluate Object Selection and Manipulation in VR? Guidelines from 20 Years of Studies. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–20.
[10] Sebastian Boring, Dominikus Baur, Andreas Butz, Sean Gustafson, and Patrick Baudisch. 2010. Touch projector: mobile interaction through video. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2287–2296.
[11] Sebastian Boring, Marko Jurmu, and Andreas Butz. 2009. Scroll, tilt or move it: using mobile phones to continuously control pointers on large public displays. In *Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7*. 161–168.
[12] Doug A Bowman and Larry F Hodges. 1997. An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments. *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, 35–ff. https://doi.org/10.1145/253284.253301
[13] Clint Carter. 2021. VR Fitness Is a Serious Workout, Seriously. https://www.mensjournal.com/health-fitness/vr-fitness-is-a-serious-workout-seriously/
[14] Gery Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1 Euro Filter: A Simple Speed-Based Low-Pass Filter for Noisy Input in Interactive Systems. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2527–2530. https://doi.org/10.1145/2207676.2208639
[15] Gery Casiez, Daniel Vogel, Ravin Balakrishnan, and Andy Cockburn. 2008. The Impact of Control-Display Gain on User Performance in Pointing Tasks. *Human-Computer Interaction* 23 (2008), 215–250. https://doi.org/10.1080/07370020802278163
[16] Yuan Chen, Keiko Katsuragawa, and Edward Lank. 2020. Understanding viewport- and world-based pointing with everyday smart devices in immersive augmented reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.
[17] Lung-Pan Cheng, Eyal Ofek, Christian Holz, Hrvoje Benko, and Andrew D Wilson. 2017. Sparse haptic proxy: Touch feedback in virtual environments using a general passive prop. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 3718–3728.
[18] Paulo Dias, Luis Afonso, Sergio Eliseu, and Beatriz Sousa Santos. 2018. Mobile Devices for Interaction in Immersive Virtual Environments. *Proceedings of the 2018 International Conference on Advanced Visual Interfaces*. https://doi.org/10.1145/3206505.3206526
[19] Tiare Feuchtner and Jörg Müller. 2018. Ownershift: Facilitating overhead interaction in virtual reality with an ownership-preserving hand space shift. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 31–43.
[20] Mar Gonzalez-Franco and Tabitha C Peck. 2018. Avatar Embodiment. Towards a Standardized Questionnaire. *Frontiers in Robotics and AI* 5 (2018), 74. https://doi.org/10.3389/frobt.2018.00074
[21] Henning Graf and Klaus Jung. 2012. The smartphone as a 3D input device: Using accelerometer and gyroscope for 3D navigation with a smartphone. *IEEE International Conference on Consumer Electronics - Berlin, ICCE-Berlin*, 254–257. https://doi.org/10.1109/ICCE-Berlin.2012.6336487
[22] Tovi Grossman and Ravin Balakrishnan. 2006. The Design and Evaluation of Selection Techniques for 3D Volumetric Displays. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology (UIST '06)*. Association for Computing Machinery, New York, NY, USA, 3–12. https://doi.org/10.1145/1166253.1166257
[23] Shangchen Han, Beibei Liu, Randi Cabezas, Christopher D Twigg, Peizhao Zhang, Jeff Petkau, Tsz-Ho Yu, Chun-Jung Tai, Muzaffer Akbay, Zheng Wang, et al. 2020. MEgATrack: monochrome egocentric articulated hand-tracking for virtual reality. *ACM Transactions on Graphics (ToG)* 39, 4 (2020), 87–1.
[24] Mark Hancock, Sheelagh Carpendale, and Andy Cockburn. 2007. Shallow-depth 3d interaction: design and evaluation of one-, two-and three-touch techniques. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1147–1156.
[25] Jeremy Hartmann, Aakar Gupta, and Daniel Vogel. 2020. Extend, Push, Pull: Smartphone Mediated Interaction in Spatial Augmented Reality via Intuitive Mode Switching. In *Symposium on Spatial User Interaction*. 1–10.
[26] Jeremy Hartmann and Daniel Vogel. 2021. An examination of mobile phone pointing in surface mapped spatial augmented reality. *International Journal of Human-Computer Studies* 153 (2021), 102662.
[27] Devamardeep Hayatpur, Seongkook Heo, Haijun Xia, Wolfgang Stuerzlinger, and Daniel Wigdor. 2019. Plane, ray, and point: Enabling precise spatial manipulations with shape constraints. In *Proceedings of the 32nd annual ACM symposium on user interface software and technology*. 1185–1195.
[28] Ken Hinckley, Randy Pausch, John C Goble, and Neal F Kassell. 1994. A Survey of Design Issues in Spatial Input. *Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology*, 213–222. https://doi.org/10.1145/192426.192501
[29] Haojian Jin, Christian Holz, and Kasper Hornbaek. 2015. Tracko: Ad-Hoc Mobile 3D Tracking Using Bluetooth Low Energy and Inaudible Signals for Cross-Device Interaction. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 147–156. https://doi.org/10.1145/2807442.2807475
[30] Nikolaos Katzakis and Masahiro Hori. 2009. Mobile phones as 3-DOF controllers: A comparative study. *8th IEEE International Symposium on Dependable, Autonomic and Secure Computing, DASC 2009*, 345–349. https://doi.org/10.1109/DASC.2009.76
[31] N Katzakis and M Hori. 2010. Mobile devices as multi-DOF controllers. *2010 IEEE Symposium on 3D User Interfaces (3DUI)*, 139–140. https://doi.org/10.1109/3DUI.2010.5444700
[32] Nicholas Katzakis, Masahiro Hori, Kiyoshi Kiyokawa, and Haruo Takemura. 2011. Smartphone Game Controller.
[33] Nicholas Katzakis, Masahiro Hori, Kiyoshi Kiyokawa, and Haruo Takemura. 2012. Plane-Casting: 3D Cursor Control with a Smartphone. *Proceedings of The 3rd Dimension of CHI (3DCHI): Touching and Designing 3D User Interfaces*, 13–21.
[34] Nicholas Katzakis, Robert J Teather, Kiyoshi Kiyokawa, and Haruo Takemura. 2015. INSPECT: extending plane-casting for 6-DOF control. *Human-centric Computing and Information Sciences* 5 (2015), 22. https://doi.org/10.1186/s13673-015-0037-y
[35] Owlchemy Labs. 2016. Job Simulator. https://jobsimulatorgame.com/
[36] Joseph J LaViola, Ernst Kruijff, Ryan P McMahan, Doub A Bowman, and Ivan Poupyrev. 2017. *3D User Interfaces: Theory and Practice*. Addison Wesley Longman

Publishing Co., Inc.
[37] Jiandong Liang and Mark Green. 1994. JDCAD: A highly interactive 3D modeling system. *Computers & graphics* 18, 4 (1994), 499–506.
[38] N G Lipari and C W Borst. 2015. Handymenu: Integrating menu selection into a multifunction smartphone-based VR controller. *2015 IEEE Symposium on 3D User Interfaces (3DUI)*, 129–132. https://doi.org/10.1109/3DUI.2015.7131737
[39] Fabrice Matulic, Aditya Ganeshan, Hiroshi Fujiwara, and Daniel Vogel. 2021. Phonetroller: Visual Representations of Fingers for Precise Touch Input with Mobile Phones in VR. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–13.
[40] David C McCallum and Pourang Irani. 2009. ARC-Pad: absolute+ relative cursor positioning for large displays with a mobile touchscreen. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*. 153–156.
[41] Daniel Mendes, Fabio Marco Caputo, Andrea Giachetti, Alfredo Ferreira, and Joaquim Jorge. 2019. A survey on 3d virtual object manipulation: From the desktop to immersive virtual environments. In *Computer graphics forum*, Vol. 38. Wiley Online Library, 21–45.
[42] Mark R Mine. 1995. Virtual Environment Interaction Techniques.
[43] Roberto A Montano Murillo, Sriram Subramanian, and Diego Martinez Plasencia. 2017. Erg-O: Ergonomic Optimization of Immersive Virtual Environments. *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, 759–771. https://doi.org/10.1145/3126594.3126605
[44] Mathieu Nancel, Olivier Chapuis, Emmanuel Pietriga, Xing-Dong Yang, Pourang P Irani, and Michel Beaudouin-Lafon. 2013. High-precision pointing on large wall displays using small handheld devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 831–840.
[45] Neilda Pacquing. 2020. The Future of Work Is Now: VR Productivity Apps To Try Out - MindGlow: Train Better With VR. https://www.mindglowinc.com/blog/the-future-of-work-is-now-vr-productivity-apps-to-try-out
[46] Krzysztof Pietroszek, James R Wallace, and Edward Lank. 2015. Tiltcasting: 3D Interaction on Large Displays Using a Mobile Device. *Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology*, 57–62. https://doi.org/10.1145/2807442.2807471
[47] Ivan Poupyrev, Mark Billinghurst, Suzanne Weghorst, and Tadao Ichikawa. 1996. The go-go interaction technique: non-linear mapping for direct manipulation in VR. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*. 79–80.
[48] Ivan Poupyrev and Tadao Ichikawa. 1999. Manipulating objects in virtual worlds: Categorization and empirical evaluation of interaction techniques. *Journal of Visual Languages & Computing* 10, 1 (1999), 19–35.
[49] Jonas Schjerlund, Kasper Hornbæk, and Joanna Bergström. 2021. Ninja Hands: Using Many Hands to Improve Target Selection in VR. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–14.
[50] Om Shah. 2020. Is Virtual Reality the Future of Productivity? https://medium.com/swlh/is-virtual-reality-the-future-of-productivity-da05a2bb5b70
[51] Noah Smith. 2021. Virtual reality is starting to see actual gains in gaming. https://www.washingtonpost.com/video-games/2021/02/04/virtual-reality-future-games/
[52] M Suhail, S P Sargunam, D T Han, and E D Ragan. 2017. Redirected reach in virtual reality: Enabling natural hand interaction at multiple virtual locations with passive haptics. *2017 IEEE Symposium on 3D User Interfaces (3DUI)*, 245–246. https://doi.org/10.1109/3DUI.2017.7893363
[53] Hemant Bhaskar Surale, Aakar Gupta, Mark Hancock, and Daniel Vogel. 2019. Tabletinvr: Exploring the design space for using a multi-touch tablet in virtual reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–13.
[54] Anran Wang and Shyamnath Gollakota. 2019. Millisonic: Pushing the limits of acoustic motion tracking. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–11.
[55] C Ware and D R Jessome. 1988. Using the bat: a six-dimensional mouse for object placement. *IEEE Computer Graphics and Applications* 8 (1988), 65–70. https://doi.org/10.1109/38.20319
[56] Johann Wentzel, Greg d'Eon, and Daniel Vogel. 2020. Improving Virtual Reality Ergonomics Through Reach-Bounded Non-Linear Input Amplification. *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 1–12. https://doi.org/10.1145/3313831.3376687
[57] Curtis B Wilkes, Dan Tilden, and Doug A Bowman. 2012. 3D User Interfaces Using Tracked Multi-touch Mobile Devices, Ronan Boulic, Carolina Cruz-Neira, Kiyoshi Kiyokawa, and David Roberts (Eds.). *Joint Virtual Reality Conference of ICAT - EGVE - EuroVR*. https://doi.org/10.2312/EGVE/JVRC12/065-072
[58] Momona Yamagami, Sasa Junuzovic, Mar Gonzalez-Franco, Eyal Ofek, Edward Cutrell, John Porter, Andrew Wilson, and Martez Mott. 2021. Two-In-One: A Design Space for Mapping Unimanual Input into Bimanual Interactions in VR for Users with Limited Movement. *arXiv preprint arXiv:2108.12390* (2021).
[59] Run Yu and Doug A Bowman. 2018. Force Push: Exploring Expressive Gesture-to-Force Mappings for Remote Object Manipulation in Virtual Reality. *Frontiers in ICT* 5 (2018), 25.