Phonetroller: Visual Representations of Fingers for Precise Touch Input with Mobile Phones in VR

Fabrice Matulic
Aditya Ganeshan
Hiroshi Fujiwara
[fmatulic,aditya,hfujiwara]@preferred.jp
Preferred Networks
Tokyo, Japan

Daniel Vogel
dvogel@uwaterloo.ca
University of Waterloo
Waterloo, Canada

ABSTRACT

Smartphone touch screens are potentially attractive for interaction in virtual reality (VR). However, the user cannot see the phone or their hands in a fully immersive VR setting, impeding their ability for precise touch input. We propose mounting a mirror above the phone screen such that the front-facing camera captures the thumbs on or near the screen. This enables the creation of semi-transparent overlays of thumb shadows and inference of fingertip hover points with deep learning, which help the user aim for targets on the phone. A study compares the effect of visual feedback on touch precision in a controlled task and qualitatively evaluates three example applications demonstrating the potential of the technique. The results show that the enabled style of feedback is effective for thumb-size targets, and that the VR experience can be enriched by using smartphones as VR controllers supporting precise touch input.

CCS CONCEPTS

• **Human-centered computing** → **Interaction techniques**; **Touch screens**; **Virtual reality**; **Mobile phones**.

KEYWORDS

VR, mobile phone, touch input, visual feedback, deep learning

ACM Reference Format:
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 *CHI Conference on Human Factors in Computing Systems (CHI '21), May 8–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3411764.3445583

1 INTRODUCTION

Standard controllers for virtual reality (VR) systems integrate motion trackers, buttons, triggers, and joysticks to allow users to navigate in and interact with the VR world. Some models, like the HTC Vive controller, also include a trackpad, but because VR headsets are fully immersive, they entirely block the outside world, hence touch operations on such trackpads are limited to location-independent gestures [45, 47] and coarse directional input [3, 20]. Mobile phones, which have even larger touch-sensitive surfaces, have also been used as touch-enabled VR controllers, but similarly, interactions are limited to dragging motions for navigation [1] and tapping of large buttons [27, 29, 42] to be usable in an eyes-free manner. Precise operations such as tapping small buttons, like in a regular phone interface, are difficult because the user has no visual feedback of their fingers aiming for the targets.

Research on text entry with physical keyboards in VR has shown that visual feedback of the hands is important for typing accuracy and efficiency [11, 24, 56]. This feedback is presumably even more critical for typing on a touchscreen where the haptic feedback of the keys is absent. To support precise touch operations such as typing on a soft keyboard, the hand and fingers of the user need to be captured *before they touch* the screen and therefore a similar window into the outside world is required. This can be achieved with cameras affixed to the headset [39, 51, 55, 62] or the environment [33, 48, 52], but user and controller movements can easily cause the hands to come out of tracking range so this type of solution is not robust. Furthermore, environment-based tracking is not suitable for mobile contexts. More flexible approaches directly use advanced phone sensors. For instance, earlier models of the Samsung Galaxy line (S4 and S5) and Sony's 2012 Xperia Sola phone included finger hover detection features called respectively "Air View" [44] and "Floating Touch" [59]. These devices use self capacitance sensors with a strong signal to detect a single hovering fingertip up to 2cm above the screen. The hover-detection capability of the Galaxy S4 was used by Kim and Kim to show thumb hover points for phone typing in VR [23]. Current smartphones do not integrate any support for this style of "pre touch" interaction [17], so direct touch input with a mobile phone in VR remains a challenge today.

In this paper, we propose Phonetroller, a low-cost system to enable visual feedback for precise touch operations in virtual reality using any mobile phone with a front camera. It consists of a mount clipped to the phone, which holds a mirror above the screen to reflect the front camera so that a top view of the hand and the thumb(s) operating the device can be acquired. This view can then be integrated in VR interfaces to show finger movements over the screen for more precise interactions. We propose two feedback techniques: A shadow showing the user's hand and thumb as a semi-transparent overlay (similar to video feeds of the user's hands

CHI '21, May 8–13, 2021, Yokohama, Japan

ACM ISBN 978-1-4503-8096-6/21/05…$15.00
https://doi.org/10.1145/3411764.3445583

Figure 1: A mobile phone is used as VR controller with visual feedback of the user's thumb shown in VR

added to GUIs in Visual Touchpad [31]), and finger hover points inferred from those shadows using deep learning. Enabled by these methods, we present applications demonstrating how a phone with precise touch input can be used to operate head-up display-style interfaces and to directly interact with VR content. We evaluate the benefits of our techniques quantitatively in a controlled targeting study and quantitatively using our demonstration applications. Our results show that targeting with thumb shadows is significantly faster than with hover points sensed by self-capacitance and beats indirect input for fingernail-sized targets.

2 RELATED WORK

Our work is related to research on mobile devices, touch input, and visual feedback for interaction in VR as well as the more generic topic of around-device sensing.

2.1 Mobile Devices and Visual Feedback in Virtual Reality

The limited support for touch input in commercial VR controllers has inspired researchers to explore the potential of common mobile devices such as mobile phones and tablets as touch-enabled controllers. Benzina et al. [1] and Liang et al. [27] use mobile phones for navigation in virtual spaces, where touch is used for translation and phone motion sensors handle rotation. HandyMenu supports menu and object selection in two distinct areas of the phone screen to facilitate eyes-free use [29]. Gugenheimer et al. [12] and Lee et al. [25] place a touch-sensitive surface on the front of a head-mounted display (HMD) to enable touch input directly on the headset. These techniques do not include any pre-touch feedback [17] to help users aim for the targets so they are limited to coarse interactions like tapping large UI elements and basic dragging gestures for movement.

The importance of visual feedback for precise touch input in VR has been highlighted in work studying text entry performance. In similar experiments, Knierim et al. [24] and Grubert et al. [11] show that representations of hands typing on physical keyboards in VR lead to increased speed and fewer errors. Walker et al. propose an entirely software-based solution consisting of a virtual keyboard assistant with visual feedback of key presses and an autocorrection algorithm to help users type more accurately [56].

Flat touch screens do not have the haptic and actuation feedback of physical keycaps, so blind taps would invariably miss small targets like soft keyboard keys. Because in these cases touch down is not a reliable event for key input, eyes-free text entry techniques on touch surfaces have tried to rely on other cues such as a pressure threshold [5] or touch release [4] for input confirmation. When tracking of the thumb is available, hover points can be overlaid on the touch keyboard to provide visual feedback. Son et al. show this improves typing performance compared to validating keys on touch release [48]. HoVR-Type similarly shows hover points on a VR keyboard, which is reportedly comparable to "aim-and-shoot" methods [23]. While hover points capture vital information for better aiming, showing the whole thumb provides additional contextual detail that could potentially improve performance and overall user experience. We compare hover points and full visual feedback of the thumb in our evaluation.

With regard to hand capturing and tracking methods, almost all use cameras attached to the HMD [14, 65] or installed in the environment [2] and mainly track bare hands. TabletInVR utilises a Leap Motion device fixed to the front of the HMD to track hands interacting with a real tablet, which are rendered as 3D models in VR [51]. This allows a range of precision multitouch gestures to be supported for the manipulation of virtual objects. For precise finger tracking using reflective markers, which the above text typing studies use, both the environment and the user are instrumented. These solutions either have limited tracking range or are impractical for everyday use. HoVR-Type, on the other hand, leverages the self-capacitance sensor of a Samsung Galaxy S4 phone to track fingertips above the screen [23]. These hover sensors are only available in a few old [44, 59] and experimental [17] phone models, however, and none of the current mobile phones have such capabilities. Our system uses only the front camera, which is a standard component of smartphones. We do include a baseline with hardware hover point detection using the Air view feature of a Galaxy S5 in our evaluation for comparison. We show that targeting accuracy aided by direct visual feedback of the hand using our mirror setup outperforms hover points obtained from the self-capacitive sensor.

2.2 Around-Device Sensing

While device-based sensing to track hands was hardly considered as an option for VR applications, a variety of techniques and systems have been proposed to enable around-device gesturing and touch input in real-world contexts. Magnetips use magnets to both track fingertips and provide haptic feedback, but the technique requires instrumentation of the device and the user's finger [32]. Millimeter-wave radar, which is the technology underlying the Soli chip integrated in the Google Pixel 4 phone, promises sub-millimeter accuracy, but in practice its strength is recognising dynamic gestures not tracking absolute finger positions [28]. Acoustic signal-based tracking can use built-in microphone and speaker, but the device needs to be stable and tracking accuracy degrades with environment noise [36, 57, 67]. Wisture uses Wi-Fi signals to detect three gestures above the phone, but is also sensitive to background interference [15].

Vision-based tracking using cameras built in or attached to the mobile device has also been explored. Air+Touch [6] and Portal-ble

[41] use a depth camera mounted on the phone to track hands and fingers in mid-air. Yang et al. attach an omni-directional mirror to the front camera to enable peripheral vision around the device [63]. LucidTouch supports back-of-device multitouch input with a see-through effect for the fingers using a camera mounted behind the device to capture the hands [58]. Wong et al. detect touch gestures on a small surface of the back of the phone using a reflector for the rear camera [61]. Song et al. detect in-air gestures performed behind the phone using the unmodified rear camera of the device [49]. Symmetrisense [64] and DeepFisheye [38] track fingertips hovering over a tablet-sized touchscreen using respectively a smartphone camera capturing the reflection of the finger on the surface and a fisheye camera attached to the bottom of the screen.

Closer to our context and requirements is Yu et al.'s HandSee technique [66]. It uses a prism mirror placed on the front camera to create a stereo vision system that generates a depth image of the hand for tracking finger location and movement. The technique is applied for mid-air and around-device gestures as well as to detect gripping postures, not for visual feedback in VR. No technical evaluation of the depth error is reported, but given the field of view from the phone screen is upwards, depth estimation and hand segmentation are likely sensitive to the background. Our approach uses a mirror *above* the phone, which gives a top-down view that does not require depth estimation for the purpose of providing simple 2D visual feedback. Furthermore, the background is the phone screen and since the user does not see the physical screen in VR, we can set it to be completely green in order to robustly segment the hand and fingers via chromakeying.

3 PHONETROLLER SYSTEM

Our prototype hardware consists of a mount attached to the phone and software to capture and send the images to the VR server for processing.

3.1 Mount

To be able to capture the hand holding the phone and fingers hovering over the screen, a top-down view is ideal. Since we want to use the built-in front camera of the phone, we need to route the optical path from the camera's position on the top bezel of the phone so that the effective camera position is above the phone. We can achieve this using a combination of reflectors and lenses, but to keep our system simple and low-cost, we use a single flat mirror held above the screen by an arm attached to the phone via a clip-on holder. The arm is constructed from Plexiglas beams attached to form joints to adjust the position and angle of the mirror. In our setup, we use a 8.5×6cm acrylic mirror held 12 cm above the screen, which is sufficient to cover the entire phone and most of the user's hand holding it (Figure 3a). For the implementation and testing of our prototype, we use a Samsung Galaxy S5, which has its front camera on the top right corner, but our mount and mirror also works for other camera locations.

To track the position and orientation of the phone in our VR environment, we attach an HTC Vive tracker on top of the mirror (Figure 2a). The tracker can be easily removed if only touch input without phone tracking is needed. The mount without the tracker weighs 90g and 178g with the tracker. The Galaxy S5 weighs 145g and its screen is 5.1 inches (13cm).

Figure 2: Mount holding mirror and VR tracker above the screen. The phone shows a green background with ArUco markers to optically locate the four corners of the screen.

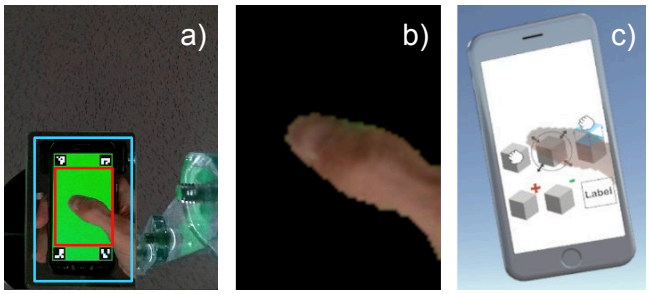


Figure 3: a) Input camera image with mirror area ROI from pre-calibration (in blue) and interactive area within the markers detected for each frame on the server (in red). b) Thumb image extracted by chromakeying. c) Thumb image overlaid in semi-transparency over virtual phone in the VR environment

3.2 Image Processing

Low latency is important to make visual feedback effective for touch input [21]. To optimise our process for speed and efficiency, we perform all heavy image processing on a Unity VR server with an i7-8700K CPU and a GeForce GTX 1080 Ti GPU.

Since the user does not see the phone when wearing the VR headset, we can display any content on the screen that will facilitate capturing and subsequent processing. We display a green screen to enable chromakey background segmentation. Additionally, we show four ArUco markers in the corners of the screen to optically locate its bounds when processing the captured image. While one marker provides sufficient information to locate those bounds, we use four markers for added redundancy and stability. Using these markers and the detected screen bounds, we perform a one-time pre-calibration to determine the mirror region (mirror ROI) in the camera image (blue rectangle in Figure 2a). We save the cropping

bounds of this mirror ROI on the phone. We are eventually interested in the phone screen region within the markers (red rectangle 3a), which we detect dynamically on the server for each frame. We do not tightly crop the interaction area at the pre-calibration stage to allow for some play in case the mount slightly moves during use or its position changes after it is removed and reattached. When running the client on the phone, the mirror ROI is cropped from the original frame and streamed as a compressed jpeg to the VR server over WiFi. Touch input data is transmitted in the same stream.

While the camera is capable of capturing at full HD resolution, we choose to capture at 240×320px as a compromise between processing efficiency and level of detail. At this resolution the pre-cropped mirror ROI is approximately 90×150px. When the server receives a frame from the phone, it first locates the ArUco markers and computes a dewarped (straightened) subimage of the area within the markers. This yields an approximately 66×92 image of the interaction area, which corresponds to 1050×1450 pixels in original screen and touch input dimensions. This dynamic screen bounds detection scheme enables us to deliver a consistent experience with stable visual feedback that is robust to slight wiggling of the mount if the user waves the phone somewhat energetically. The marker positions are continuously saved and updated so that even if the user temporarily covers one or more markers with their thumb or palm, the system is still able to dewarp the input image.

In the rectified and cropped image of the interactive area, we extract the thumb image from the green background using chromakeying (Figure 2b). Depending on the desired representation in the application, the thumb image can be left as is or converted to a coloured mask with drawn contours for enhanced effect (Figure 4d). If realism is not paramount, a smaller thumbprint may be preferable to reduce the "fat thumb" effect. The thumb shadow can be thinned by eroding the mask contour by a few pixels.

Finally, the thumb image is rendered in semi-transparency over the desired VR element, such as the virtual screen of a 3D phone model (Figure 2c), thus providing visual feedback of mid-air thumb movements to the user without occluding the content underneath. Alternatively, instead of representing the entire thumb, we can show the hovering fingertip as a point, similar to Air View [44] and Floating Touch [59]. To obtain this hover point without any hardware sensor, we apply deep learning on the thumb image. This approach is described in section 6.

Despite our best efforts to reduce latency, the transmission of image data over WiFi and its processing incur some delays. With a high-speed camera, we measured a 175ms end-to-end latency between the actual thumb position and the corresponding image appearing in VR. For comparison, we also measured the end-to-end latency of the S5's Air View hover point detection to be 350ms, i.e. twice as large. With regard to frame rates, the server receives 30 frames per second and a scene in Unity containing just a canvas on which the thumb images are rendered updates at 65 fps on average. When hover point inference with deep learning is activated, the frame rate drops to 20 fps and latency rises to 600ms, as the load on the GPU is considerably increased. Higher fps and lower latency can likely be achieved with a more powerful server.

3.3 Touch calibration

Prior work established that there is a slight mismatch between the contact point perceived by users when tapping a touchscreen and the actual touch point coordinates registered by the device [19]. This discrepancy is exacerbated with the unconventional viewing angle of the front camera through the mirror, which differs from usual viewpoints when using the phone normally. Without correction, the touch point appears at an unintuitive location with respect to the thumb mask, sometimes outside its contour. We therefore include a touch calibration tool similar to standard calibration procedures used for touch devices, with crosshairs that need to be successively tapped. The original touch point is then corrected using a perspective transform based on the four calibration points provided by the user.

4 APPLICATIONS

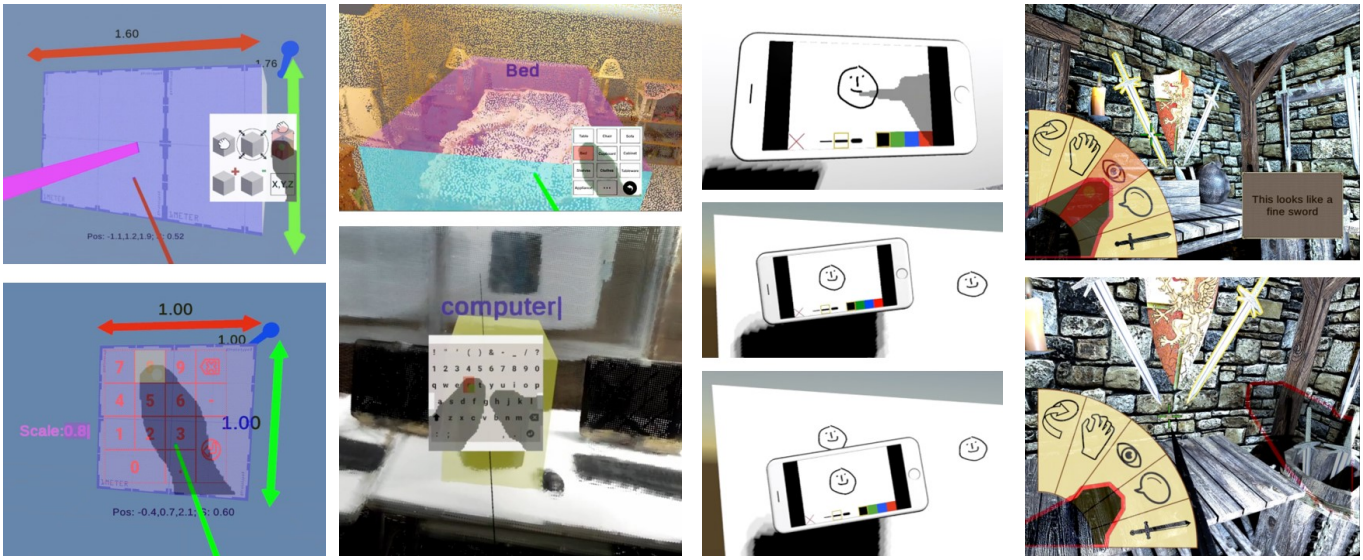
We create several demonstration applications to show different potential uses of Phonetroller with its visual feedback and hover detection capabilities. The ability for the user to see their hand and fingers in VR creates a design space that comes close to that of a phone used in augmented reality (AR) where the user's hand is directly visible and precise touch input is possible [34, 54, 68]. The VR environment and the phone-based thumb tracking further allow us to consider more immersive scenarios where visual feedback of the thumb can be made constantly available regardless of whether the user is looking at their hand or not.

Our exploration includes interaction and representation styles with the phone used as pointing device and simple touch interface, unimanual and bimanual interaction (using a second VR controller in combination with the phone), small opaque or full-screen transparent HUDs, text entry, selecting using head orientation or finger hovering and pen input. Our applications tackle four scenarios each covering some of these aspects: block modelling and a variation thereof for 3D data annotation, sketching with a pen, and touch gaming. These applications are also used in the qualitative part of our evaluation.

All applications are shown in the accompanying explanation video.

4.1 3D Modelling Using Block Structure

3D modelling is a popular scenario for VR and our first demonstrator is a block-based modelling tool somewhat similar to Surale et al's prototype for TabletInVR [51], itself inspired by Minecraft-style games. In our application, users create and manipulate cubes using the phone, where objects are selected and manipulated by pointing the phone at them (raycasting) and tools selected by tapping buttons in the UI (Figure 4a). We support basic operations such as creating/deleting blocks, moving and scaling them and pulling/pushing individual faces with mid-air dragging movements. Additionally, and contrary to most VR modelling tools that we are aware of, we provide the possibility to enter specific numeric values for dimensions and transform operations via a touch numeric keypad, e.g. to enter a specific scale factor or x,y,z position in the 3D space. Tool selection and typing of numeric values are facilitated by the thumb shadow, which is shown as a semi-transparent overlay on the UI as described above.



(a) 3D Block Modelling (b) Point Cloud Labelling (c) Sketch and Paste (d) Touch+VR Game

Figure 4: Demonstration applications

In its standard setting, the phone is represented as a virtual 3D phone (without the mount) and the user can see the UI with the modelling tools rendered as a texture on the virtual screen. This UI is also shown as a fixed HUD in a corner of the HMD's viewport. The HUD allows the user to view the tools without having to look down at the virtual phone. Since we can easily detect if the user is holding their thumb above the phone screen, we can use the presence and absence of the thumb in the camera's view as condition to respectively show and hide the HUD. This allows the user to summon the tools only when desired and otherwise keep a full open view of the VR space. The user can switch to a centre-screen HUD with only icon outlines by bringing the controller close to the HMD (Figure 4a bottom). This HUD style further helps to simultaneously focus on content and tools.

We support an alternative bimanual method to point at and manipulate blocks using a second controller. In this configuration, the user holds a standard VR controller in their other hand to point at objects and the phone is only used to select tools with touch. In this case, the VR tracker attached to the top of the mirror can be removed, thus reducing the total weight of the phone. The phone can further be held at a comfortable position that minimises arm fatigue. This bimanual option allows separate hands to undertake pointing and tool selection roles, but it also requires the user to coordinate the actions of both hands. This problem is extensively studied in work on bimanual interaction [7, 13, 18, 37]. We compare these two different interaction methods in the qualitative part of our user evaluation.

4.2 Annotation of 3D Point Clouds

We create a variation of our block-based modelling tool to demonstrate its potential for the labelling of 3D data such as point clouds used to train machine learning models. We believe this is an area in which VR can make a valuable contribution, as already demonstrated by a few prototypes [50, 60]. In this version of the application, blocks are used as bounding volumes to select points in the cloud. Specifically, blocks are created and shaped so as to enclose the points belonging to a particular object, such as a table or a chair in an indoor scene. These bounding volumes are semi-transparent to maintain visibility of the points contained inside (Figure 4b). Labels for these points can be assigned by selecting from a list of predefined labels or by entering custom labels with a keyboard.

We use the annotator version of the block manipulation tool as one of the test applications in our evaluation.

4.3 Sketch and Paste

Pen interaction, such as sketching and note-taking, is another category that requires visual feedback for precise input. Sketch and Paste is a rudimentary sketching application that allows users to doodle on the screen of the virtual phone and then use the device like a rubber stamp to paste their creations on a virtual wall (Figure 4c). In a multi-user context, this wall could be a shared graffiti surface for artists or a board to attach virtual sticky notes in a brainstorming meeting. We think the phone with its tile-like shape particularly lends itself to these rubber stamp and sticky note metaphors and therefore makes the VR experience in these types of scenarios more compelling. Furthermore, since we are in VR and have full control over the size of the virtual phone and the feedback texture, we can increase the size of the virtual device to enhance the visibility of the strokes and the pen shadow. In our demonstration, we double the size of the virtual phone compared to the real device, thus creating a tablet-size virtual sketching canvas.

In addition to the toolbar at the bottom of the canvas containing buttons to change colour and stroke width, we support an eraser function, where the user can rub the screen with their finger instead of the pen to erase. We distinguish between pen and touch input

using a classification neural network, which is described in section 6.

Sketch and Paste is also part of the application test set of our evaluation.

4.4 Mobile Touch+VR Gaming

We envisage scenarios where phones can be used in combination with VR solely as touch devices, i.e. without any position tracking. This use case is particularly relevant in spaces where room is limited and arm/hand movement is constrained so that pointing with a controller is not possible, e.g. in a small cramped room, on a plane etc. In these contexts, where the user is mostly stationary and body movement is limited to the head, touch input is the main vector of interaction. While this may slightly downgrade the VR experience, it also reduces arm fatigue, in particular the "gorilla arm" effect when holding up arms and hands to point with the controllers for a long time.

To explore this touch-driven style of interaction, we create a mock-up scene of an adventure game, where the user holds the phone in landscape orientation with both hands on each side, like a handheld game console. In this setting, the touch input space is mapped to the entire HMD, meaning the overlay with the thumb images spans the full viewport and the user can interact with the VR content with touch, much like a mobile game but augmented with VR. Contrary to traditional mobile games, however, the system sees the user's thumbs above the screen. We can take advantage of this by detecting whether the user is holding their thumb above the screen to trigger an action, as in the block-modelling UI. With both hands used for interaction in this case, we can determine *which* thumb is moved over the phone screen by checking the presence of thumb pixels on each side of the overlay image. In our game mock-up, we use the on-screen appearance of the left thumb as a trigger to summon a fan-shaped menu (Figure 4d). This menu contains a selection of icons corresponding to different game actions: teleport, take, look at, talk and attack. The right thumb is then used to execute the action on target objects in the game. For instance, after the user selects the attack icon from the menu with the left thumb, a weapon is drawn. This weapon can then be wielded by moving and tapping with the right thumb.

For navigation, we include teleport points materialised as blue halos on the ground. Users can virtually hop between these locations by pointing at the teleport points and tapping the corresponding action in the menu. Default tapping actions also exist so that teleporting can be simply performed by directly tapping the blue halos.

In another departure from traditional mobile games, where object selection can only be performed by tapping, we support selecting/highlighting interactive elements in the VR space by hovering over them, similar to a mouse cursor. This is made possible by our deep learning hover point estimator, which derives a cursor point from the thumb image. Alternatively, we can use a more VR-centric pointing method, which selects objects using head orientation, i.e. based on what the user is looking at. In this case, the cursor is materialised as a crosshair in the centre of the viewport, similar to first-person shooter games. We compare thumb hovering and crosshair pointing as selection methods in our evaluation.

5 USER STUDY

To assess the benefits of visual feedback for phone interaction in VR we conduct a three-part user evaluation. The first part is a controlled experiment aimed at quantitatively evaluating touch input precision against a baseline and alternative techniques. The second part focuses on more subjective aspects related to different interaction styles and experiences enabled by our VR applications. A final third part gathers thumb image data from the participant to later train deep learning models for hover point inference.

5.1 Part 1: Controlled Tasks

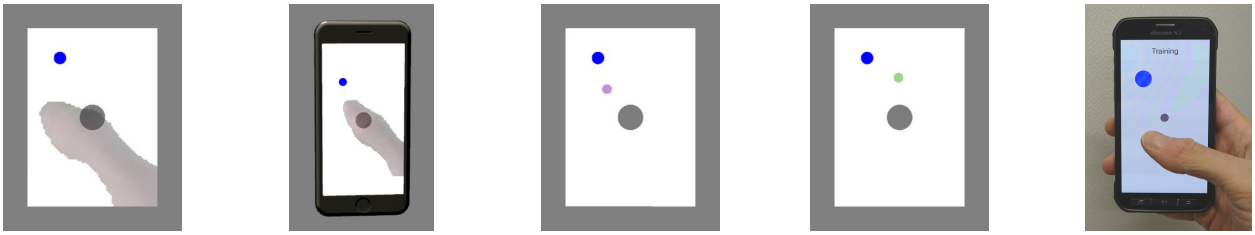
Similar to related studies on text entry in VR, we seek to experimentally validate the claim that visual feedback enables or facilitates increased input precision and efficiency. Our focus is not limited to text entry, so we use a generic target selection task inspired by Lehmann and Kipp [26]. Circular targets of different sizes are successively shown at different locations on the screen and participants tap them as rapidly and precisely as they can. An estimate of efficiency can be obtained by measuring the time taken to successfully tap targets, with precision determined by error rates (number of targets missed). The current target is shown as a blue circle and the next target as a grey circle to help participants prepare for the next move. In all conditions, a 80px-wide green dot appears at the touch location when the user contacts the screen (touch point).

5.1.1 Conditions. Using this task, we evaluate five conditions: three variations of visual feedback in VR, a no-feedback VR baseline and a real phone no VR baseline (Figure 5):

- HUD-SHADOW: The thumb image is shown as an overlay over a fixed white rectangle (HUD) on which the targets are displayed.
- PHONE-SHADOW: Same condition, but with a virtual phone.
- HUD-HOVERPOINT: Uses the Air View feature of the S5 to show a hover point as a purple dot of diameter 80px when in range.
- HUD-BLIND: No pre-touch feedback is provided. Participants first blindly touch the screen, then, when the green dot cursor appears, drag it inside the target and finally lift their finger to confirm. This technique serves as a basis of comparison when no feedback is available. Note that this input technique is not a complete substitute for trigger-on-tap methods in case touch down or dragging is explicitly used as input event to perform an action, e.g. sketching with a pen, gesturing, activation of maintained modes, dragging used for panning or scrolling etc.
- NOVR-REALPHONE: Trials are performed directly on the phone without VR and without the mirror mount attached to the phone. This baseline condition represents the lower bound with presumably the lowest execution times and number of errors. The goal is to determine how closely the VR conditions can approach this ideal situation.

To keep study sessions to a reasonable length of time for our participants, our conditions include only one instance with the virtual phone as a variation of thumb shadow, our main visual feedback technique. While we do not test the virtual phone with the other VR conditions, we hope to gain some insights into the possible influence of the "virtual form factor".

5.1.2 Study Design. In order for the task conditions to be equal for each tested technique, the number, position and size of the targets



(a) HUD-SHADOW (b) PHONE-SHADOW (c) HUD-HOVERPOINT (purple dot is hover point) (d) HUD-BLIND (green dot is touch point. Target validated on touch up) (e) NOVR-REALPHONE

Figure 5: The five conditions of the controlled tasks

need to be fixed. Since this controlled task is only one part of the study, we limit the number of different sizes to two and locations to the four corners and the centre of the screen, i.e. five locations. We choose 104px (≈6.2mm) for the small size (diameter) of our circle targets as it corresponds to the width of a key on the soft keyboard. Our large size is double that length, i.e. 208px (≈12.3mm), which is just under the average thumbnail width [22] and is close to the width of the eroded thumb tip in the mask image. These two types of targets can be viewed as proxies for keyboard keys and UI buttons.

We construct trial blocks with our five locations and two target sizes, where users have to tap from corners to the centre of the phone and vice versa as well as across diagonals and edges. Our sequence for a given target size consists of 16 such trials. A block starts with a sequence for large targets and is followed by the same sequence with small targets. If a target is missed, the previous target, which is also the starting point of the current trial, needs to be retapped. This ensures that we end up with the same number of successful trials and prevents participants from racing through the tasks. We record completion times and targeting errors for the quantitative analysis of performance and precision.

The main study task includes two blocks as well as a short warm-up set of 5 trials for each size, which are not logged. This gives a total of 5 trials × 2 sizes + (16 trials × 2 sizes × 2 blocks) = 74 trials, of which only the last 64 are counted.

Our study adopts a within-subjects design with each participant experiencing all five techniques. The order of the techniques is changed for each participant following a balanced Latin square rule. Before starting the main trials, participants are given the opportunity to train as long as they wish with each technique. Before the first VR technique is used, the touch calibration procedure described above is performed. Participants may repeat the calibration if during training they feel that the touch point (represented by the green dot) appears at undesirable or inconsistent locations with respect to the thumb shadow. All VR conditions use corrected touch points to ensure consistency.

Participants are told to mainly hold the phone with a single hand, but considering its weight, we allow them to support the device with their other hand if they wish. We attach a holder ring on the back of the phone in which participants can slide their middle or index finger for additional stability. Participants are instructed to perform the tasks as quickly as possible while avoiding errors. All touch input is to be performed using the thumb of the phone-holding hand in a sitting position. We do not evaluate two-thumb, multitouch, or landscape-mode input in the controlled tasks.

5.2 Part 2: Application Experience

The subjective, qualitative part of the study uses the three demonstration applications described previously: the point cloud annotator, Sketch and Paste and the touch game mock-up. No specific task is given and participants can freely experience the applications (in standing or sitting position), but the experimenter encourages them to try out all the supported features. For the point cloud annotator and the game mock-up, where two pointing methods exist (respectively pointing with another controller vs phone, and crosshair vs hovering thumb), the experimenter manually switches between the two alternatives and asks participants which one they prefer and why. General feedback about the applications is collected during use as well as in a post-experiment interview.

Deep learning models to distinguish pen and touch in Sketch and Paste and to infer hover points for pointing in the game mock-up are integrated in those two demos. The models are trained with the data of two pilot testers only, so accuracy is not optimal, but since the focus is to evaluate the general user experience, this is not a major concern.

5.3 Part 3: Data Gathering

In the final 5 minutes of the study, participants provide training data for the inference of hover points using the deep neural network described in section 6. They do so by dragging their thumb slowly across the interactive area of the screen (without wearing the HMD) while camera images and touch points are recorded. We use touch points as an approximation of hover points as we found Air View hover points to be less reliable.

5.4 Participants

We recruited 19 participants: 17 male, 2 female of mean age 33 years (SD=8), all right-handed[1]. Nine participants had no prior experience with VR, seven a little and two declared owning VR headsets. No participant reported vision problems or other impairments. At the

[1]The experiments were carried out in the summer of 2020 during the Covid-19 pandemic. Work-from-home restrictions made it difficult to recruit a large number of diverse participants.

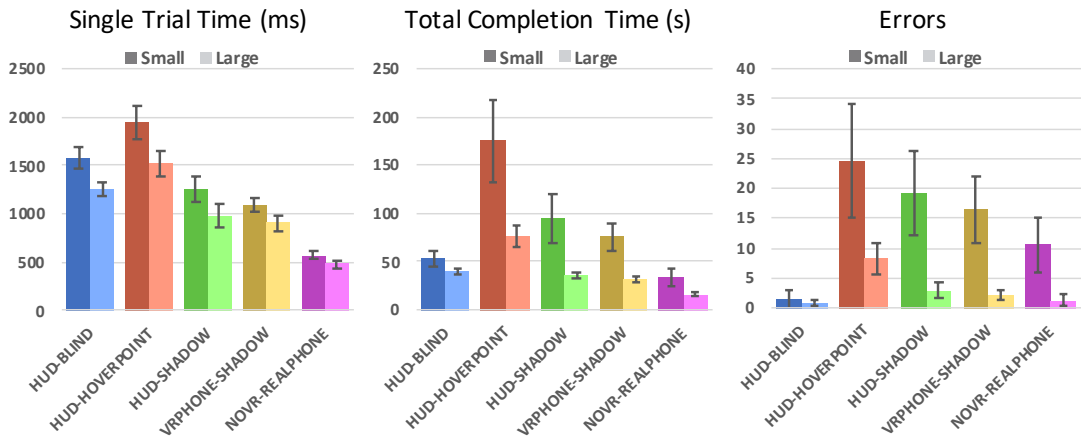


Figure 6: Results of controlled tapping tasks divided into large and small targets. Error bars represent 95% confidence intervals.

end of a session, people were given sweets as a thank-you for their participation.

5.5 Results

A session took on average 1 hour and 10 minutes.

5.5.1 Controlled Tasks. We consider the following three metrics for our quantitative analysis:

- *Single Trial Time*: The time between two successful taps on start and target circles. We use the median value of all considered trials for the aggregate value to limit the influence of outliers. This metric gives us an idea of a representative speed people would achieve in the given condition.
- *Total Completion Time*: The time to complete a full set of trials, i.e. including the time to rectify errors (moving a step back to tap the previous target again).
- *Errors*: The number of times a target is missed.

An ANOVA on the data did not show any significant learning effect between the two blocks so we include both for our analyses. We perform a 2-way ANOVA on Technique × Target Size and obtain significant main effects as well as interactions with all p<.001. We therefore consider the two target sizes separately to compare the conditions using our metrics. The results are shown in Figure 6.

We notice a high variability, especially regarding the number of errors. ANOVAs on all three metrics for both target sizes exhibited main effects (p<.001) so we conducted post hoc tests with Bonferroni correction for all measures. Due to the large number of pairwise comparisons we report p values only for a selection of cases that we directly compare. When omitted, the significance or non-significance of the differences based on the 0.05 threshold is clear (values <0.001 or close to 1).

Overall, the worst condition was HUD-HOVERPOINT. It had longer *Single Trial Time* and *Total Completion Time* for all targets and higher *Errors* for large targets (the differences for small targets are not significant). Participants confirmed their dislike of the technique in their feedback, commenting that the hover point was jumpy and unstable. It also often did not align with the actual touch point on touch down.

The best condition, unsurprisingly, was NOVR-REALPHONE. It had the lowest *Single Trial Time* and *Total Completion Time*, despite a slightly but significantly higher number of *Errors* than HUD-BLIND. Trial times for this baseline are on average half those of the next best technique, PHONE-SHADOW. We also expected HUD-BLIND to exhibit the highest accuracy because of the possibility to adjust the touch point after tapping. For small targets, HUD-BLIND is the only method with significantly lower total *Errors* than other conditions. All other techniques exhibit non-significant differences between their error rates for small targets. Overall, *Errors* are relatively high for small targets, including for NOVR-REALPHONE. For larger targets, HUD-BLIND is only significantly more precise than HUD-HOVERPOINT and the two shadow methods, albeit with borderline p-values (0.049 for HUD-SHADOW and 0.041 for PHONE-SHADOW).

With regard to *Total Completion Time*, for large targets HUD-BLIND is significantly outperformed by PHONE-SHADOW (p=0.004), but the difference with HUD-SHADOW is not significant (p=0.765). The situation is somewhat reversed for small targets with HUD-BLIND being almost significantly faster than HUD-SHADOW (p=0.052) but not so when compared to PHONE-SHADOW (p=0.152). When looking at individual trial times, however, both shadow techniques are significantly faster for all targets. This suggests that with practice, shadow techniques are a better option regardless of target size. This intuition is further supported by one of our pilot testers with longer exposure to shadow feedback who completed the whole task with HUD-SHADOW in less than 50s with just 3 errors. Moreover, the participant, who spent the most time practising and recalibrating, made the lowest number of errors for PHONE-SHADOW (2) and achieved a relatively low completion time (73.8s).

When comparing the two shadow conditions, the charts seemingly show a slight advantage for PHONE-SHADOW over HUD-SHADOW, but only the trial times of small targets were significantly different (p=0.015). Participants were asked for their subjective preferences between the virtual phone and the fixed HUD and the former was slightly favoured over the latter (11 vs 8 participants). This slight edge is likely due to the fact that manipulating a virtual phone feels closer to a real phone experience. Preferences may of course vary depending on the task context.

5.5.2 Application Experience. Our applications did not require speedy and precise tapping so there were no major performance concerns when testing our demos. One exception is when participants used the keyboard to enter custom labels for the bounding volumes in the point cloud annotator. This proved harder than using a touch keyboard without VR, but felt less demanding than the controlled trials, since the keys were all close to each other. Two of our participants noticed that if they closed one eye they could type more comfortably, so the difficulty to type may be partially due to focus problems resulting from an improperly adjusted headset or blurry lenses.

A third of the participants mentioned that the phone was heavy and its centre of gravity with the mount was high. Two of these participants said that it affected their precision and the way they gripped the device. This points to the need for a lightweight tracking solution both for hovering fingers and for the phone.

The UI with the tools to manipulate blocks occupied about 2/3rd of the phone screen making button tapping possible without changing grip. Participants quickly learned how to move and modify blocks using maintained button presses on the phone like pressing a physical button on a conventional controller to grasp objects. Tapping on a touch screen to do these operations especially did not feel out of place for participants with previous VR experience.

The annotator requires attention on both the selection blocks and the UI so their proximity to each other is paramount. This is an example where a HUD is more practical than a virtual phone object anchored to the position of the real device. Opinions were evenly split among participants when asked if they preferred a full-screen HUD with just outlines of the tool symbols or a more opaque UI in a corner of the viewport. This justifies the need for an action to switch between the two HUD types.

With regard to pointing preferences, a slight majority of participants favoured using the phone for both pointing and tool selection over using a second controller to point (11 vs 8). People who preferred bimanual interaction indicated the separation of pointing and touching roles and the ability to use a lighter controller to point as benefits. Participants who would rather use only their dominant hand for everything pointed out the burden of coordinating the movement of two hands in the bimanual case.

Opinions similarly varied regarding the two options to point and highlight objects in the game demo. 11 participants preferred hovering with their thumb to select, while 5 liked the crosshair better. The main reason given for preferring the thumb is the need to move the head to align the crosshair with content, which may cause neck strain and fatigue. On the other hand, aiming with the crosshair was reportedly more precise. One participant stated they would use the thumb to teleport since the teleport halos are on the ground (and therefore require more effort to look at) and the crosshair to point at content at eye level. Again, these different aiming preferences call for flexibility and the support of different pointing methods in applications.

Regarding the game experience, overall, participants found it enjoyable and appreciated the comparatively rich interaction possibilities given the limited demand on body movement, especially when using the thumb to interact. People quickly adopted familiar game interaction patterns, selecting actions in the menu, tapping on highlighted objects and dragging the screen to wield the sword. One participant, however, observed that limiting hand interaction to an invisible 2D plane somewhat reduced the sense of immersion in the VR world. This is indeed one aspect that differs from traditional VR experiences, where hands and controllers are fully embodied in the 3D space.

As for Sketch and Paste, this was the most liked demonstration, as it was seen as a playful novel experience. Participants enjoyed using the phone as a rubber stamp to paste their sketches on a virtual wall. One participant thought this application would be even more effective with a real tablet (rather than artificially doubling the size of the phone in VR). A minor issue pointed out by two participants was that the pen was only visible when its shadow was within the interactive area of the phone. This meant they occasionally had to fumble when trying to find the phone screen with the pen. This disruption might also have been somewhat compounded by the size mismatch between the real and virtual phone.

An issue that affected our applications used in landscape mode with the phone held in both hands is that participants sometimes covered the camera with their left thumb. This resulted in a large shadow suddenly appearing on the overlay, but participants quickly reacted by removing their thumb. Nevertheless, this somewhat constrains gripping possibilities for the left hand. This problem would be aggravated with devices whose front camera is in the centre of the bezel.

5.6 Discussion

Perhaps the clearest finding of the controlled experiments is that Air View is not a reliable cue for precise touch input. The hover points are too unstable, limited to a short tracking range above the screen and do not always match with the touch point on contact. Whether these issues are due to the S5 or representative of self-capacitive sensors in general is unclear.

Our evaluation confirms that Phonetroller with thumb shadows as visual feedback can facilitate touch input in VR, especially when targets are at least as large as fingernails. For smaller targets and more precise aiming, some practice is required to learn to correctly estimate where the touch point will appear based on the thumb shadow. This also underlines the importance of the touch calibration, which determines the relative location of the corrected touch point. One problem is that the thumb contacts differently depending on the touch location on the screen. In the lower corner towards the palm of the hand, the thumb bends to touch whereas it is fully stretched when aiming for the opposite far corner of the display. Since the corrected touch point is obtained by interpolating from the four corner points input during calibration, it may appear at unexpected locations in these intermediate zones if the calibration is not performed with this in mind. Many participants realised these adverse effects after the first calibration and adjusted how they aimed for the crosshairs in subsequent recalibrations to ensure consistency of the touch point location within the thumb shadow.

The above problem is symptomatic of 2D feedback with lack of depth information. A full 3D representation of the hand would very likely mitigate these issues, but constructing an accurate 3D representation of the user's hand using data captured from built-in phone sensors is not easy, especially if this needs to be done at high

speed and low latency. Phonetroller is a low-cost, low-engineering solution, which improves on the status quo, but with its limitations.

One impediment that can be addressed more straightforwardly is weight and balance. While we can only hope that VR trackers will become smaller and lighter, the use of a convex mirror with a wide viewing angle instead of a flat mirror should make it possible to reduce the height of the reflector over the phone and hence lower the centre of gravity. This would also enable wider coverage and support for larger devices such as tablets. It might be possible to further reduce the weight by utilising 6DoF localisation and tracking techniques using only the phone's sensors (camera, IMU and depth sensors for newer models) [9, 16, 35, 40], which would eliminate the need for the VR tracker. We plan to explore these solutions in the future.

A technique that fared relatively well in the controlled tasks is blind tapping with validation on touch up (HUD-BLIND). This technique, which does not use any pre-touch feedback, is the most precise and requires very little time to get used to (participants felt comfortable using it after only minimal training). However, it has a cost. As mentioned previously, compared to touch down, tap-drag-release as an interaction sequence monopolises two extra input events that cannot be used for other actions. Several interactions in our applications such as tool activation with maintained button press, sketching and weapon wielding would not work with this input method. Typing would theoretically be possible but entering keys on touch up is unusual and therefore may feel unnatural to users. Secondly, confirm-on-release is not incompatible with visual feedback. In fact, pre-touch visual aids could be used in combination with confirm-on-release to initially better point at targets and significantly reduce subsequent dragging adjustments. Two participants remarked that trying to blindly aim to be as close as possible to targets in order to minimise dragging demanded additional cognitive effort. Thumb images or hover points would eliminate that mental burden and doubtlessly decrease targeting time [23].

Regarding applications, we believe we were able to demonstrate as of yet unexplored potential of mobile devices as touch-enabled VR controllers. Phonetroller can effectively turn mobile phones into tracked remote controls featuring rich touch-operated toolsets. Precise input enabled by pre-touch visual feedback allows UIs to densely pack touch-operated widgets, similar to regular phone interfaces. While we did not compare Phonetroller with standard VR controllers for menu selection, prior work shows that touch is more efficient than raycasting for target acquisition [46] so we believe Phonetroller is a compelling system for menu operation.

As for our initial exploration of hybrid mobile touch + VR games borrowing interaction patterns from both worlds, we believe it is a promising paradigm, despite the reduced sense of immersion caused by plane-bound touch input. Perhaps depth dimension can be added by actively using the hovering distance (which we do not currently estimate) or touch pressure. Games could also be specifically designed for these constraints. We think this is an interesting direction for future work.

Another category with a seemingly promising future is pen-based VR applications. Pen input is another type of interaction that seems difficult to support without some form of representation of the pen. We note a few recent works showing effective use of pens in VR [8, 10]. They all rely on separate external tracking to locate pen position in 3D space. Logitech recently developed an enterprise VR stylus for SteamVR [30], but it is relatively bulky and expensive. Our system only provides a 2D texture for the feedback, but it is low-cost, uses only the phone for sensing and works with standard touch pens.

Finally, one participant suggested that phones in VR could also be used with their regular apps. For instance, consider a scenario where a user is engaged in a VR activity but needs to temporarily access their phone close by, e.g. to check a notification, quickly reply to an email etc. Phonetroller would allow the user to use their phone without leaving the VR world, i.e. without removing the headset, making context switches less costly. Another scenario would be to maintain privacy in public spaces. Since the content on the physical screen of the phone does not need to be visible, mobile apps could be used entirely in VR, i.e. safe from surrounding prying eyes.

6 HOVER-POINT INFERENCE AND INPUT METHOD CLASSIFICATION

We introduce the Deep Neural Network (DNN) model utilised for two tasks: inference of the 2D screen coordinates corresponding to the hover position of the fingertip, and recognition of the input tool (pen vs. finger touch).

We initially experimented with traditional computer vision (CV) techniques to estimate hover points, but quickly established they were not able to adequately capture different thumb shapes and grips. The touch point appears at different locations within the thumb shadow (crucially *not* at the thumb's extremity, which is what finger detection techniques for mid-air gesturing typically use), depending on how much the thumb bends to reach different parts of the phone screen. In the end, rather than trying to hand-build a model for these nuances, we opted for a DNN approach, which has shown its superiority and practicality for many similar CV tasks.

We treat hover-point inference as a keypoint-estimation problem. The DNN, denoted as f, takes the extracted thumb image (Figure 3b) ($I \in R^{H\times W\times 1}$) as input and predicts a 2D probability heatmap ($P_{pred} \in R^{H\times W\times 1}$), where H and W represent the height and width respectively. Our DNN architecture is inspired from U-net [43]. Similar to Thompson et al. [53], we convert the hover-point annotation (x, y) into a 2D Gaussian probability map $P_{GT} \in R^{H\times W\times 1}$ (with a fixed sigma $\sigma = 2.5$, and mean $\mu = (x, y)$), and utilise the Binary Cross Entropy loss between the prediction P_{pred} and ground truth map P_{GT} to train the network. For the task of input-tool recognition, which we employ in the Sketch and Paste application (see Section 4.3), the feature encoded by 5 Convolutional layers is passed through a small fully-connected network to perform two-way classification: pen vs touch (trained with cross entropy loss). Further details of our DNN with the full network architecture are provided in the Appendix.

6.1 Experiments

Due to the higher latency of our learning-based approach, we only provide a theoretical analysis of the performance of our neural

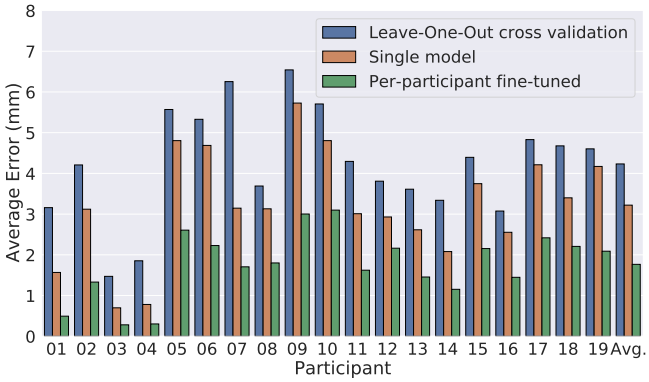


Figure 7: Average error achieved by our models (with different training strategies) for each participant. Fine-tuning the model for each-participant is found to be the optimal solution.

network and leave the practical validation with more powerful hardware to future work.

We use the images gathered in part 3 of the study as input data for our experiments. Our dataset consists of 208617 top-view thumb images from 19 participants. For each participant, we use 70% randomly selected data as training data, and the rest as validation data.

We compare the performance of different training strategies: (a) Single model: A single model trained with all the data, (b) Per-participant fine-tuned model: The model trained with all data is fine-tuned for each participant on their data, and (c) Leave-one-out cross-validation model: For each participant, we train the model on all data but their own. In Figure 7, we report the performance for these strategies. We found strategy (b) to be the optimal choice with a mean Average Error of 1.72 mm across participants. We observed that some of the images contained noisy thumb masking (due to reflections) and hover-point annotation errors (due to lag in registration), so accuracy may have been affected by these issues.

Based on this evaluation, we posit that applying DNNs for hover-point inference is promising. Our model is able to achieve relatively high accuracy, at a speed of 20 frames per-second on our system. With the rapid increase of GPU performance, larger and more reliable networks can be considered in the future.

7 CONCLUSION

We presented Phonetroller, a novel low-cost system to use mobile phones as VR controllers with visual feedback of the hand in VR to facilitate precise touch input. We proposed capturing hovering thumbs with the front-facing camera of the phone through the reflection of a mirror placed above the screen. We showed how the captured images of the thumb can be overlaid as semi-transparent layers over touch-operated HUD interfaces in VR. We further used deep learning on these images to infer fingertip hover points, which can be used for feedback or hover-based input. We demonstrated the potential of Phonetroller via three applications illustrating various uses of precise touch input in VR: 3D block modelling with a variation for point cloud annotation, pen sketching and mobile touch+VR gaming. A user evaluation confirmed fingernail-sized targets can be precisely hit with the help of thumb shadows, and validated the quality of our applications as novel and rich VR experiences. We believe each of the scenarios we have begun to explore merit further in-depth investigation and we hope we have inspired other researchers to consider mobile touch-driven VR for future research.

REFERENCES

[1] Amal Benzina, Arindam Dey, Marcus Toennis, and Gudrun Klinker. 2012. Empirical Evaluation of Mapping Functions for Navigation in Virtual Reality Using Phones with Integrated Sensors. In *Proceedings of the 10th Asia Pacific Conference on Computer Human Interaction* (Matsue-city, Shimane, Japan) *(APCHI '12)*. Association for Computing Machinery, New York, NY, USA, 149–158. https://doi.org/10.1145/2350046.2350078

[2] Verena Biener, Daniel Schneider, Travis Gesslein, Alexander Otte, Bastian Kuth, Per Ola Kristensson, Eyal Ofek, Michel Pahud, and Jens Grubert. 2020. Breaking the Screen: Interaction Across Touchscreen Boundaries in Virtual Reality for Mobile Knowledge Workers. *IEEE Transactions on Visualization and Computer Graphics* 26, 12 (2020), 3490–3502.

[3] Costas Boletsis and Stian Kongsvik. 2019. Controller-based Text-input Techniques for Virtual Reality: An Empirical Comparison. *International Journal of Virtual Reality (IJVR)* 19, 3 (Oct. 2019), 2–15. https://doi.org/10.20870/IJVR.2019.19.3.2917

[4] Sabah Boustila, Thomas Guégan, Kazuki Takashima, and Yoshifumi Kitamura. 2019. Text Typing in VR Using Smartphones Touchscreen and HMD. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, New York, NY, USA, 860–861. https://doi.org/10.1109/VR.2019.8798238

[5] Sibo Chen, Junce Wang, Santiago Guerra, Neha Mittal, and Soravis Prakkamakul. 2019. Exploring Word-Gesture Text Entry Techniques in Virtual Reality. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI EA '19)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3290607.3312762

[6] Xiang 'Anthony' Chen, Julia Schwarz, Chris Harrison, Jennifer Mankoff, and Scott E. Hudson. 2014. Air+touch: Interweaving Touch & in-Air Gestures. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (Honolulu, Hawaii, USA) *(UIST '14)*. Association for Computing Machinery, New York, NY, USA, 519–525. https://doi.org/10.1145/2642918.2647392

[7] Lawrence D. Cutler, Bernd Fröhlich, and Pat Hanrahan. 1997. Two-Handed Direct Manipulation on the Responsive Workbench. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics* (Providence, Rhode Island, USA) *(I3D '97)*. Association for Computing Machinery, New York, NY, USA, 107–114. https://doi.org/10.1145/253284.253315

[8] Tobias Drey, Jan Gugenheimer, Julian Karlbauer, Maximilian Milo, and Enrico Rukzio. 2020. VRSketchIn: Exploring the Design Space of Pen and Tablet Interaction for 3D Sketching in Virtual Reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376628

[9] Wei Fang, Lianyu Zheng, and Xiangyong Wu. 2017. Multi-sensor based real-time 6-DoF pose tracking for wearable augmented reality. *Computers in Industry* 92-93 (2017), 91 – 103. https://doi.org/10.1016/j.compind.2017.06.002

[10] Travis Gesslein, Verena Biener, Philipp Gagel, Daniel Schneider, Per Ola Kristensson, Eyal Ofek, Michel Pahud, and Jens Grubert. 2020. Pen-based Interaction with Spreadsheets in Mobile Virtual Reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, New York, NY, USA, 361–373.

[11] J. Grubert, L. Witzani, E. Ofek, M. Pahud, M. Kranz, and P. O. Kristensson. 2018. Effects of Hand Representations for Typing in Virtual Reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, New York, NY, USA, 151–158.

[12] Jan Gugenheimer, David Dobbelstein, Christian Winkler, Gabriel Haas, and Enrico Rukzio. 2016. FaceTouch: Enabling Touch Interaction in Display Fixed UIs for Mobile Virtual Reality. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 49–60. https://doi.org/10.1145/2984511.2984576

[13] Yves Guiard. 1987. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of motor behavior* 19, 4 (1987), 486–517.

[14] Shangchen Han, Beibei Liu, Randi Cabezas, Christopher D. Twigg, Peizhao Zhang, Jeff Petkau, Tsz-Ho Yu, Chun-Jung Tai, Muzaffer Akbay, Zheng Wang, Asaf Nitzan, Gang Dong, Yuting Ye, Lingling Tao, Chengde Wan, and Robert Wang. 2020. MEgATrack: Monochrome Egocentric Articulated Hand-Tracking for Virtual Reality. *ACM Trans. Graph.* 39, 4, Article 87 (July 2020), 13 pages. https://doi.org/10.1145/3386569.3392452

[15] Mohamed Abdulaziz Ali Haseeb and Ramviyas Parasuraman. 2017. Wisture: RNN-based Learning of Wireless Signals for Gesture Recognition in Unmodified Smartphones. arXiv:1707.08569 [cs.HC]

[16] Keisuke Hattori and Tatsunori Hirai. 2020. Inside-out Tracking Controller for VR/AR HMD Using Image Recognition with Smartphones. In *ACM SIGGRAPH 2020 Posters* (Virtual Event, USA) *(SIGGRAPH '20)*. Association for Computing Machinery, New York, NY, USA, Article 23, 2 pages. https://doi.org/10.1145/3388770.3407430

[17] Ken Hinckley, Seongkook Heo, Michel Pahud, Christian Holz, Hrvoje Benko, Abigail Sellen, Richard Banks, Kenton O'Hara, Gavin Smyth, and William Buxton. 2016. Pre-Touch Sensing for Mobile Interaction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 2869–2881. https://doi.org/10.1145/2858036.2858095

[18] Ken Hinckley, Koji Yatani, Michel Pahud, Nicole Coddington, Jenny Rodenhouse, Andy Wilson, Hrvoje Benko, and Bill Buxton. 2010. Pen+ touch= new tools. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. Association for Computing Machinery, New York, NY, USA, 27–36.

[19] Christian Holz and Patrick Baudisch. 2011. Understanding Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 2501–2510. https://doi.org/10.1145/1978942.1979308

[20] H. Jiang and D. Weng. 2020. HiPad: Text entry for Head-Mounted Displays Using Circular Touchpad. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, New York, NY, USA, 692–703.

[21] Ricardo Jota, Albert Ng, Paul Dietz, and Daniel Wigdor. 2013. How Fast is Fast Enough? A Study of the Effects of Latency in Direct-Touch Pointing Tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 2291–2300. https://doi.org/10.1145/2470654.2481317

[22] Jin Woo Jung, Kwang Seog Kim, Jun Ho Shin, Yu Jin Kwon, Jae Ha Hwang, and Sam Yong Lee. 2015. Fingernail configuration. *Archives of plastic surgery* 42, 6 (2015), 753.

[23] Y. R. Kim and G. J. Kim. 2017. HoVR-Type: Smartphone as a typing interface in VR using hovering. In *2017 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, New York, NY, USA, 200–203.

[24] Pascal Knierim, Valentin Schwind, Anna Maria Feit, Florian Nieuwenhuizen, and Niels Henze. 2018. Physical Keyboards in Virtual Reality: Analysis of Typing Performance and Effects of Avatar Hands. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–9. https://doi.org/10.1145/3173574.3173919

[25] Jihyun Lee, Byungmoon Kim, Bongwon Suh, and Eunyee Koh. 2016. Exploring the Front Touch Interface for Virtual Reality Headsets. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI EA '16)*. Association for Computing Machinery, New York, NY, USA, 2585–2591. https://doi.org/10.1145/2851581.2892344

[26] Florian Lehmann and Michael Kipp. 2018. How to Hold Your Phone When Tapping: A Comparative Study of Performance, Precision, and Errors. In *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces* (Tokyo, Japan) *(ISS '18)*. Association for Computing Machinery, New York, NY, USA, 115–127. https://doi.org/10.1145/3279778.3279791

[27] Hai-Ning Liang, Yuwei Shi, Feiyu Lu, Jizhou Yang, and Konstantinos Papangelis. 2016. VRMController: An Input Device for Navigation Activities in Virtual Reality Environments. In *Proceedings of the 15th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry - Volume 1* (Zhuhai, China) *(VRCAI '16)*. Association for Computing Machinery, New York, NY, USA, 455–460. https://doi.org/10.1145/3013971.3014005

[28] Jaime Lien, Nicholas Gillian, M. Emre Karagozler, Patrick Amihood, Carsten Schwesig, Erik Olson, Hakim Raja, and Ivan Poupyrev. 2016. Soli: Ubiquitous Gesture Sensing with Millimeter Wave Radar. *ACM Trans. Graph.* 35, 4, Article 142 (July 2016), 19 pages. https://doi.org/10.1145/2897824.2925953

[29] N. G. Lipari and C. W. Borst. 2015. Handymenu: Integrating menu selection into a multifunction smartphone-based VR controller. In *2015 IEEE Symposium on 3D User Interfaces (3DUI)*. IEEE, New York, NY, USA, 129–132.

[30] Logitech. 2020. Logitech VR Ink Pilot Edition. https://www.logitech.com/en-roeu/promo/vr-ink.html. Accessed: 2020-09-15.

[31] Shahzad Malik and Joe Laszlo. 2004. Visual Touchpad: A Two-Handed Gestural Input Device. In *Proceedings of the 6th International Conference on Multimodal Interfaces* (State College, PA, USA) *(ICMI '04)*. Association for Computing Machinery, New York, NY, USA, 289–296. https://doi.org/10.1145/1027933.1027980

[32] Jess McIntosh, Paul Strohmeier, Jarrod Knibbe, Sebastian Boring, and Kasper Hornbæk. 2019. Magnetips: Combining Fingertip Tracking and Haptic Feedback for Around-Device Interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300638

[33] Tim Menzner, Travis Gesslein, Alexander Otte, and Jens Grubert. 2020. Above Surface Interaction for Multiscale Navigation in Mobile Virtual Reality. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, New York, NY, USA, 372–381.

[34] Alexandre Millette and Michael J McGuffin. 2016. DualCAD: integrating augmented reality with a desktop GUI and smartphone interaction. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*. IEEE, New York, NY, USA, 21–26.

[35] Peter Mohr, Markus Tatzgern, Tobias Langlotz, Andreas Lang, Dieter Schmalstieg, and Denis Kalkofen. 2019. TrackCap: Enabling Smartphones for 3D Interaction on Mobile Head-Mounted Displays. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3290605.3300815

[36] Rajalakshmi Nandakumar, Vikram Iyer, Desney Tan, and Shyamnath Gollakota. 2016. FingerIO: Using Active Sonar for Fine-Grained Finger Tracking. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 1515–1525. https://doi.org/10.1145/2858036.2858580

[37] Russell Owen, Gordon Kurtenbach, George Fitzmaurice, Thomas Baudel, and Bill Buxton. 2005. When It Gets More Difficult, Use Both Hands: Exploring Bimanual Curve Manipulation. In *Proceedings of Graphics Interface 2005* (Victoria, British Columbia) *(GI '05)*. Canadian Human-Computer Communications Society, Waterloo, CAN, 17–24.

[38] Keunwoo Park, Sunbum Kim, Youngwoo Yoon, Tae-Kyun Kim, and Geehyuk Lee. 2020. DeepFisheye: Near-Surface Multi-Finger Tracking Technology Using Fisheye Camera. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 1132–1146. https://doi.org/10.1145/3379337.3415818

[39] Ken Pfeuffer, Benedikt Mayer, Diako Mardanbegi, and Hans Gellersen. 2017. Gaze + Pinch Interaction in Virtual Reality. In *Proceedings of the 5th Symposium on Spatial User Interaction* (Brighton, United Kingdom) *(SUI '17)*. Association for Computing Machinery, New York, NY, USA, 99–108. https://doi.org/10.1145/3131277.3132180

[40] Alwin Poulose and Dong Seog Han. 2019. Hybrid Indoor Localization Using IMU Sensors and Smartphone Camera. *Sensors* 19, 23 (2019), 5084.

[41] Jing Qian, Jiaju Ma, Xiangyu Li, Benjamin Attal, Haoming Lai, James Tompkin, John F. Hughes, and Jeff Huang. 2019. Portal-Ble: Intuitive Free-Hand Manipulation in Unbounded Smartphone-Based Augmented Reality. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New York, NY, USA, 133–145. https://doi.org/10.1145/3332165.3347904

[42] RiftCat. 2020. VRidge Controller. https://play.google.com/store/apps/details?id=com.riftcat.vridgecontroller. Accessed: 2020-08-21.

[43] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi (Eds.). Springer International Publishing, Cham, 234–241.

[44] Samsung. 2013. How does Air view work? https://www.samsung.com/global/galaxy/what-is/air-view/. Accessed: 2020-08-21.

[45] Shizof. 2019. Touch Gesture VR. https://www.nexusmods.com/skyrimspecialedition/mods/27815. Accessed: 2020-08-21.

[46] Shaishav Siddhpuria, Sylvain Malacria, Mathieu Nancel, and Edward Lank. 2018. *Pointing at a Distance with Everyday Smart Devices*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3173574.3173747

[47] José Somolinos. 2017. How to design common/basic interactions in VR. https://blog.prototypr.io/how-to-design-common-basic-interactions-in-vr-f958cf160cfc?gi=e3daf7483cd1. Accessed: 2020-08-21.

[48] Jeongmin Son, Sunggeun Ahn, Sunbum Kim, and Geehyuk Lee. 2019. Improving Two-Thumb Touchpad Typing in Virtual Reality. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI EA '19)*. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3290607.3312926

[49] Jie Song, Gábor Sörös, Fabrizio Pece, Sean Ryan Fanello, Shahram Izadi, Cem Keskin, and Otmar Hilliges. 2014. In-Air Gestures around Unmodified Mobile Devices. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (Honolulu, Hawaii, USA) *(UIST '14)*. Association for Computing Machinery, New York, NY, USA, 319–329. https://doi.org/10.1145/2642918.2647373

[50] Jonathan Dyssel Stets, Yongbin Sun, Wiley Corning, and Scott W. Greenwald. 2017. Visualization and Labeling of Point Clouds in Virtual Reality. In *SIGGRAPH Asia 2017 Posters* (Bangkok, Thailand) *(SA '17)*. Association for Computing Machinery, New York, NY, USA, Article 31, 2 pages. https://doi.org/10.1145/3145690.3145729

[51] 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* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300243

[52] Jonathan Taylor, Lucas Bordeaux, Thomas Cashman, Bob Corish, Cem Keskin, Toby Sharp, Eduardo Soto, David Sweeney, Julien Valentin, Benjamin Luff, Arran

Topalian, Erroll Wood, Sameh Khamis, Pushmeet Kohli, Shahram Izadi, Richard Banks, Andrew Fitzgibbon, and Jamie Shotton. 2016. Efficient and Precise Interactive Hand Tracking through Joint, Continuous Optimization of Pose and Correspondences. *ACM Trans. Graph.* 35, 4, Article 143 (July 2016), 12 pages. https://doi.org/10.1145/2897824.2925965

[53] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. 2015. Efficient object localization using Convolutional Networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, New York, NY, USA, 648–656.

[54] Rishi Vanukuru, Amarnath Murugan, and Jayesh Pillai. 2020. Dual Phone AR: Using a Second Phone as a Controller for Mobile Augmented Reality. In *26th ACM Symposium on Virtual Reality Software and Technology* (Virtual Event, USA) *(UIST '20 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 117–119. https://doi.org/10.1145/3379350.3416139

[55] Spyros Vosinakis and Panayiotis Koutsabasis. 2018. Evaluation of visual feedback techniques for virtual grasping with bare hands using Leap Motion and Oculus Rift. *Virtual Reality* 22, 1 (2018), 47–62.

[56] James Walker, Bochao Li, Keith Vertanen, and Scott Kuhl. 2017. Efficient Typing on a Visually Occluded Physical Keyboard. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 5457–5461. https://doi.org/10.1145/3025453.3025783

[57] Wei Wang, Alex X. Liu, and Ke Sun. 2016. Device-Free Gesture Tracking Using Acoustic Signals. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking* (New York City, New York) *(MobiCom '16)*. Association for Computing Machinery, New York, NY, USA, 82–94. https://doi.org/10.1145/2973750.2973764

[58] Daniel Wigdor, Clifton Forlines, Patrick Baudisch, John Barnwell, and Chia Shen. 2007. Lucid Touch: A See-through Mobile Device. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (Newport, Rhode Island, USA) *(UIST '07)*. Association for Computing Machinery, New York, NY, USA, 269–278. https://doi.org/10.1145/1294211.1294259

[59] Wikipedia. 2012. Sony Xperia sola - Wikipedia. https://en.wikipedia.org/wiki/Sony_Xperia_sola. Accessed: 2020-08-21.

[60] Florian Wirth, Jannik Quchl, Jeffrey Ota, and Christoph Stiller. 2019. PointAtMe: Efficient 3D Point Cloud Labeling in Virtual Reality. In *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, New York, NY, USA, 1693–1698.

[61] Pui Chung Wong, Hongbo Fu, and Kening Zhu. 2016. Back-Mirror: Back-of-Device One-Handed Interaction on Smartphones. In *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications* (Macau) *(SA '16)*. Association for Computing Machinery, New York, NY, USA, Article 13, 2 pages. https://doi.org/10.1145/2999508.2999512

[62] Peter Wozniak, Oliver Vauderwange, Avikarsha Mandal, Nicolas Javahiraly, and Dan Curticapean. 2016. Possible applications of the LEAP motion controller for more interactive simulated experiments in augmented or virtual reality. In *Optics Education and Outreach IV*, G. Groot Gregory (Ed.), Vol. 9946. International Society for Optics and Photonics, SPIE, Bellingham WA 98227-0010 USA, 234 – 245. https://doi.org/10.1117/12.2237673

[63] Xing-Dong Yang, Khalad Hasan, Neil Bruce, and Pourang Irani. 2013. Surround-See: Enabling Peripheral Vision on Smartphones during Active Use. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. Association for Computing Machinery, New York, NY, USA, 291–300. https://doi.org/10.1145/2501988.2502049

[64] Chungkuk Yoo, Inseok Hwang, Eric Rozner, Yu Gu, and Robert F. Dickerson. 2016. SymmetriSense: Enabling Near-Surface Interactivity on Glossy Surfaces Using a Single Commodity Smartphone. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 5126–5137. https://doi.org/10.1145/2858036.2858286

[65] Shahrouz Yousefi, Mhretab Kidane, Yeray Delgado, Julio Chana, and Nico Reski. 2016. 3D gesture-based interaction for immersive experience in mobile VR. In *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE, New York, NY, USA, 2121–2126.

[66] Chun Yu, Xiaoying Wei, Shubh Vachher, Yue Qin, Chen Liang, Yueting Weng, Yizheng Gu, and Yuanchun Shi. 2019. HandSee: Enabling Full Hand Interaction on Smartphone with Front Camera-Based Stereo Vision. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300935

[67] Sangki Yun, Yi-Chao Chen, Huihuang Zheng, Lili Qiu, and Wenguang Mao. 2017. Strata: Fine-Grained Acoustic-Based Device-Free Tracking. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services* (Niagara Falls, New York, USA) *(MobiSys '17)*. Association for Computing Machinery, New York, NY, USA, 15–28. https://doi.org/10.1145/3081333.3081356

[68] Fengyuan Zhu and Tovi Grossman. 2020. BISHARE: Exploring Bidirectional Interactions Between Smartphones and Head-Mounted Augmented Reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376233