Above-Screen Fingertip Tracking and Hand Representation for Precise Touch Input with a Phone in Virtual Reality

Fabrice Matulic
fmatulic@preferred.jp
Preferred Networks
Japan

Taiga Kashima
tkashima@preferred.jp
Preferred Networks
Japan

Deniz Beker
denizbekerjp@gmail.com
Preferred Networks
Japan

Daichi Suzuo
suzuo@preferred.jp
Preferred Networks
Japan

Hiroshi Fujiwara
hfujiwara@preferred.jp
Preferred Networks
Japan

Daniel Vogel
dvogel@uwaterloo.ca
University of Waterloo
Canada

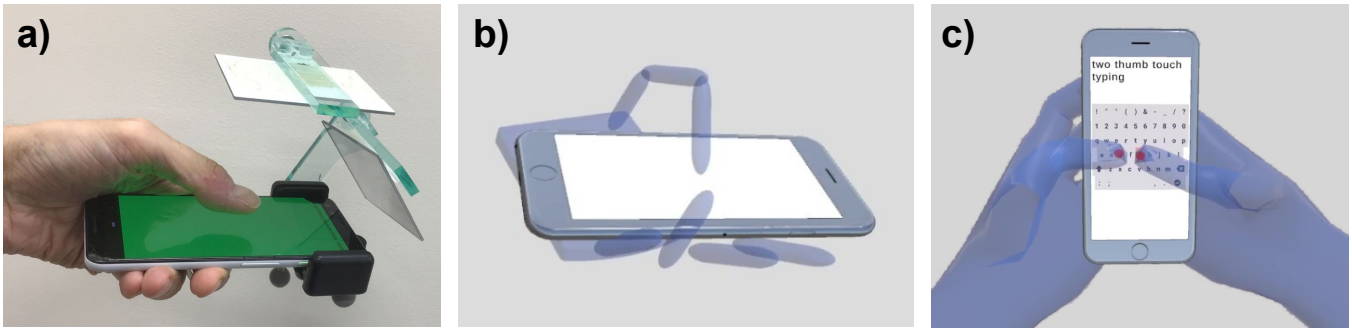


Figure 1: Phone-based fingertip-tracking system and hand representations: (a) a custom mount with two mirrors captures the hand operating a phone from two different angles, 3D fingertip positions estimated with deep learning are used to control different hand representations, e.g. (b) an abstract hand with stick-like fingers, (c) 3D hands with markers on fingertips to facilitate precise touch input such as typing on a phone keyboard.

ABSTRACT

Interacting with the touchscreen of a mobile phone in virtual reality (VR) is challenging because users cannot see their fingers when aiming for targets. We propose using two mirrors reflecting the front camera of the phone and a purpose-built deep neural network to infer the 3D position of fingertips above the screen. Network training is self-supervised after only a few hundred initial labelled images and does not require any external sensor. The inferred fingertip positions can be used to control different hand models and objects in VR. Controlled experiments evaluate tracking performance for single-finger touch input, and compare several 3D hand representations with a flat 2D overlay used in previous work. The results confirm the suitability of our fingertip tracker to aid precise tapping of small targets on the phone screen and provide insights about the effect of various hand representations on control and presence. Finally, we provide several application examples showing how 3D fingertip input can complement and extend phone-based touch interaction in VR.

GI '24, June 03–06, 2024, Halifax, NS, Canada

ACM ISBN 979-8-4007-1828-1/24/06
https://doi.org/10.1145/3670947.3670961

CCS CONCEPTS

• **Human-centered computing** → **Interaction tech**.

KEYWORDS

virtual reality, hand pose estimation

ACM Reference Format:
Fabrice Matulic, Taiga Kashima, Deniz Beker, Daichi Suzuo, Hiroshi Fujiwara, and Daniel Vogel. 2024. Above-Screen Fingertip Tracking and Hand Representation for Precise Touch Input with a Phone in Virtual Reality. In *Graphics Interface (GI '24), June 03–06, 2024, Halifax, NS, Canada.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3670947.3670961

1 INTRODUCTION

The widespread availability of mobile phones has sparked interest in using them as controllers in virtual reality (VR) [7, 11, 23, 30, 46]. A key challenge has been how to track and represent users' hands and fingers operating the phone, which is critical for precise touch input, such as typing and tapping buttons. Some VR devices can track hands and render them as 3D models, but current commercial systems cannot robustly determine the pose of a hand holding an

object [19, 37] like a phone[1]. One workaround is to superimpose a camera feed of the segmented hand (selective passthrough) [5, 34], but the result is often noisy with visual artefacts that do not blend well with computer-generated graphics, hands beyond the camera view cannot be captured and hand geometry cannot be modified. Another approach exploits specialised capacitive touchscreens that can track finger hover [26], but phones with such capabilities are rare and currently not sold as new. Moreover, this style of hover tracking is imprecise [34] and feedback is limited to a 2D cursor for a single fingertip.

We propose a technique that uses the front camera of a standard mobile phone to track thumb and index fingertips above the touchscreen. Our approach is inspired by MirrorTablet [29] and Phonetroller [34], both of which use a mirror to capture images of the hand with the front camera and project them as flat overlays for 2D visual feedback. We significantly extend this idea by using *two* mirrors capturing the hand from distinct perspectives (Figure 1a) and a deep neural network to estimate 3D fingertip positions. Training data for the network is generated with a differentiable rendering pipeline using only a few labelled images initially to improve hand segmentation. No external sensor or user instrumentation is required. With the inferred 3D fingertip position, diverse virtual hand representations can be created with different levels of realism and optimisations for precise touch input.

In a controlled experiment, we evaluate tapping and tracing performance, and compare several 3D hand representations with Phonetroller's flat 2D overlay. Our results show that hand models with enhanced fingertip visualisation, such as stick-like fingers, virtual hands with fingertip markers and 3D cursors outperform the 2D overlay for tapping tasks. Finally, we present a few examples of applications and touch+above screen interactions enabled by our fingertip tracking technique to inspire deeper investigations of that design space.

In summary, we make three contributions:

- *A deep learning technique* consisting of a novel auto-labelling pipeline for 3D fingertip estimation using mostly self-supervised finger annotation. Only the front camera and a double-mirror mount are required to capture RGB images for input. No external sensor is needed for training or inference and the technique is highly adaptable with minimal annotation cost, making it potentially widely accessible.
- *Design insights* about the effect of various hand representations on tapping and tracing performance as well as users' sense of control and presence for VR scenarios involving precise touch input with a mobile phone.
- *Application examples* showing how robust phone-based fingertip tracking can be used beyond simple visual feedback for standard touch operations, including head-up interfaces, double raycasting, and novel touch and above-screen finger interaction for phone-controlled VR games.

This work is based on our CHI 2023 Late-Breaking Work[35].

2 RELATED WORK

We review existing approaches that focus on around-phone interaction outside of VR, the use of smartphones as VR controllers, hand tracking with touch and object interaction, and hand representations in VR environments.

2.1 Around-Device Interaction Outside of VR

Several around-device hand and finger tracking techniques have been proposed to extend touch interaction with a mobile phone to mid-air gestures in non-VR contexts. Commonly, finger detection is performed via a camera, for example with the phone's built-in rear camera for back-of-device gesturing [50], a mounted depth camera [12], a dedicated hand-tracking sensor [41], and capacitive proximity sensing [14, 21]. Some works use mirrors to extend the camera's viewing range. For instance, Back-Mirror enables touch input on the back of a phone using a small mirror reflecting the rear camera[57] and Surround-See supports peripheral vision through an omnidirectional mirror placed on the front camera [59]. Those systems either require an additional sensor, do not track hands above the screen, or are designed for coarse gesturing and thus are not suitable for high-precision 3D tracking of fingertips. MirrorTablet uses a mirror above a tablet to capture the user's hand in order to provide visual feedback for remote collaboration [29]. The hand is not tracked and is just shown as a 2D overlay on the collaborator's device. HandSee places a prism with a mirror on its hypotenuse on the front camera to create stereoscopic images through double reflection for the detection of finger gestures above the screen [61]. No measured depth errors for fingertip tracking are reported, but the proximity of the two viewpoints of the two virtual cameras created by the prism likely makes it difficult to detect small movements in the depth dimension with high precision. Coverage is also limited in the upper area of the phone screen near the prism, especially for cameras with low field of view angles.

Besides techniques using cameras to detect hand gestures around mobile phones, there have been different approaches using magnets [38], millimetre-wave radar [31], GSM [65] and acoustic signals [40, 56, 62]. Those solutions either require extra sensors or are sensitive to environmental noise.

In summary, none of the above techniques are suitable for the VR scenario we consider, which requires precise and robust finger tracking above the phone screen without additional sensors.

2.2 Smartphones as VR Controllers

Smartphones have been considered for use as VR controllers, initially without visual feedback of the hand in the VR scene. User interfaces in those cases are designed for "blind" control, e.g. using large touch areas, swiping and phone tilting [7, 11, 23, 30, 46].

When visual feedback of the hand operating the phone is available, traditional touch UIs can be used, as users can aim more precisely. Son et al. [49] and HoVR-Type [26] track thumbs above the screen to aid typing using respectively a motion capture system and the hover detection feature of a Samsung Galaxy S4. Those solutions are not practical for most VR contexts as they rely on costly or discontinued sensing hardware. Bai et al. [5] and Zhang et al. [64] develop augmented virtuality systems that capture hands manipulating a phone with a depth camera mounted on the headset and render them over a virtual phone in VR, which is aligned with the real device. The rendering exhibits several artefacts due to the imperfect colour-based segmentation of the skin, the fingertips are not tracked and the camera on the headset must directly face the hands without occlusion to be able to capture them. Because

[1]The accompanying video demonstrates issues when tracking a hand holding a phone.

of those limitations, it is not possible to replace the virtual hands and phone with completely different representations, nor can virtual hands be seamlessly repositioned or redirected for remapped interfaces [33].

Phonetroller uses the front camera of the phone to capture images of hands manipulating the device through the reflection of a downward-facing mirror mounted above the screen [34]. Only the portion of the hand that is directly above the screen is shown as a 2D texture overlay in VR, which provides limited visual feedback with a greatly reduced sense of depth. Furthermore, since fingers are not tracked, the possibilities for alternative hand and finger representations are significantly restricted. Our system expands the idea of using a reflector for hand capturing by adding a second mirror with a different orientation. This allows us to train a deep learning model that can infer the 3D position of the fingertip to control 3D hand models or other virtual objects, whose appearances are in harmony with the VR scene. Furthermore, our technique does not require per-user touch calibration. We use Phonetroller as one of our evaluation baselines to compare direct 2D visualisation of the real hand with virtual 3D representations for precise touch input.

2.3 Hand-Object Pose Estimation

Current VR headsets and hand-tracking sensors can track hands with relatively high spatial accuracy, but even the best systems exhibit an average positional error of more than 1cm for fingertip detection [1, 47], which is twice the width of a key on a typical phone keyboard [18, 45]. Zhu et al. show that precision can be increased by dynamically repositioning the hand model upon finger contact on the phone screen to match the touch point [67], but those realignments cause the hand to suddenly jump at each touch, and the technique was only evaluated in a fixed position. Furthermore, HMD hand trackers are mainly designed for bare hands and their performance degrades with hands holding objects [19, 37]. In TabletInVR, which investigates the use of a tablet device for modelling in VR [51], hands are tracked by a Leap Motion on the headset and materialised in VR, but the authors note that the sensor is sensitive to screen reflections and certain hand angles, which leads them to design their gestures around those constraints. Furthermore, touch input precision is not evaluated in that work.

Computer vision techniques and datasets have been proposed to estimate the pose of hands holding and manipulating objects without markers [8, 20, 28, 52, 53, 58], but they rely on cameras on the headset or in the environment so tracking performance may decrease if the hands move to suboptimal viewing angles and distances. Furthermore, these techniques have not been applied to the specific scenario of precisely tracking a hand manipulating a mobile phone.

2.4 Hand Representation in VR

Prior work has investigated the effect of hand representations on task performance and users' sense of embodiment. For pick and place tasks, Argelaguet et al. establish that sense of agency is higher for more abstract hand representations, but sense of ownership is stronger for realistic virtual hands [3]. Schwind et al. [48], Jung et al. [22] and Yoon et al. [60] further confirm the importance of hand realism to ensure a high level of presence in VR.

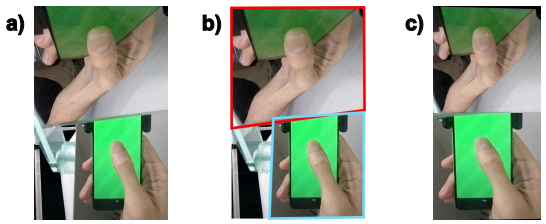


Figure 2: Preprocessing of the captured frames on the phone. a) Source image; b) the two regions of interest (ROI) with the two mirrored views; c) perspective transformation of the two ROIs into square images (concatenated back in a single image when streamed from the phone).

Grubert et al. compare different hand representations for typing on a physical keyboard in VR and find that visualising only the fingertips or showing a direct video inlay delivered the best performance [17]. In a similar study, Knierim et al. do not observe any significant performance impact of hand appearance, but record the lowest workloads for realistic hands, especially for inexperienced typists [27]. Bai et al. compare three different visualisation styles of hands operating a phone and while no significant performance difference was found, a fully opaque visualisation of the hands was deemed more immersive by participants compared to semi-transparent or white hands [5]. However, these results could be due to the quality of the rendering, which uses raw point clouds rather than a clean synthetic mesh of a 3D hand model. Contrary to those findings, Van Veldhuizen and Yang show that semi-transparent hand models increase performance in tasks requiring high precision [55].

While those studies provide some valuable insights, none of them investigated the effect of virtual hand representations with different shapes on touch precision using a mobile phone. Our work seeks to at least partially fill that gap.

3 SELF-SUPERVISED 3D FINGERTIP DETECTION

Our goal is to track fingertips above and on the phone screen in the local 3D space of the device with high precision using only the front camera as sensor. With the phone position (tracked by other means beyond the scope of this work), we can then obtain fingertip positions in world space to control virtual hand models or other 3D objects.

3.1 Two-Mirror System

For high tracking precision and coverage on all three dimensions using RGB images as input, multi-view capturing is preferable [10]. A common multi-view approach is to use stereo vision, but systems with a small distance between the two cameras and similar angles, such as the prism used in HandSee [61], require perfect calibration and stereo matching to yield a low depth error. This is difficult to achieve with self-built low-cost equipment. We therefore consider a two-view setup with two mirrors mounted on the phone that reflect the front camera at two different positions and angles: One near-vertical mirror placed close to the camera, which produces a rear view, and a second mirror placed parallel to the screen above it, which produces a top view (Figure 1a). Both mirrors are positioned

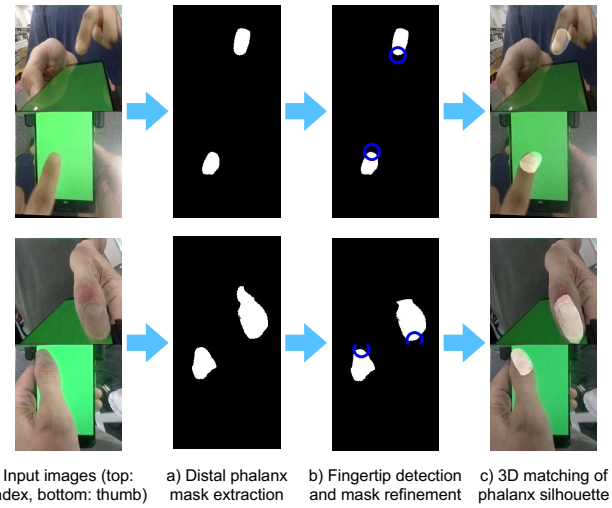


Figure 3: Data creation pipeline consisting of three steps: From the preprocessed input images, a) extracting mask regions of the fingers' distal phalanges, b) determining the 2D fingertip location on the contour of the masks (blue circles) and refining the masks, c) obtaining the 3D position and orientation of the phalanges via differentiable rendering based on the masks and the 2D fingertip position.

so that they each reflect the whole phone screen and the immediate space above it on half of the vertical pixel space in the captured frame (Figure 2a). Compared to Phonetroller [34], which uses only a single top view mirror, essentially we add a rear mirror, which, while slightly increasing the weight of the mount, provides an additional view for 3D pose estimation.

3.1.1 Frame Preprocessing. In a preprocessing step, the two mirror views are cropped from the source camera frames according to manually defined regions of interest (Figure 2b), transformed into two square images (Figure 2c), which form the input of the deep learning pipeline responsible for estimating the 3D pose of thumb and index finger tips.

3.1.2 Camera Calibration. We treat each mirror view as a virtual camera and perform standard calibration procedures using a mini ChArUco board and ArUco markers displayed on the phone screen to obtain intrinsic and extrinsic parameters for each virtual camera. Those parameters allow us to transform pixel coordinates into 3D points in the phone's coordinate system, whose origin we set to be the middle of the top bezel of the device.

3.2 Data Creation

To train a neural network that estimates the 3D position of fingertips based on RGB images, ground truth data with images of fingers labelled with the corresponding 3D positions of their tips is required. These labels can be automatically determined using an external sensor, such as a high-precision depth camera or a motion capture system. However, such equipment is typically expensive or cumbersome to set up. We propose a data creation method that does not rely on any external sensor and only requires training images obtained from people casually manipulating the phone.

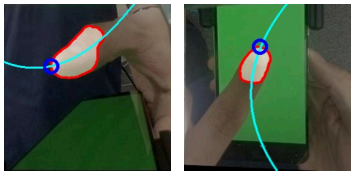


Figure 4: Examples of fingertip detection. The tip position is heuristically approximated by determining the intersection point between the phalanx mask contour and a fitted polynomial curve on the expected side of the fingertip (blue circle).

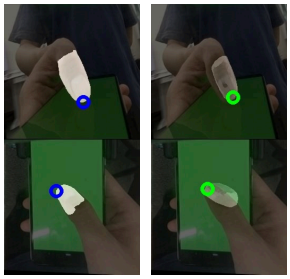


Figure 5: Segmented distal phalanx with detected tip (2D) and the corresponding 3D mesh position obtained with differentiable rendering.

The data creation pipeline consists of three steps (Figure 3): (a) Segmentation of the distal phalanx in the pair of input images; (b) detecting the 2D fingertip on the segmentation mask; and (c) using differentiable rendering to obtain the best-fitting 3D pose based on the rendered silhouette and the 2D fingertip on the mask.

3.2.1 Distal Phalanx Segmentation. There are no public datasets or models optimised for the segmentation of distal phalanges, so we adapt the pre-trained salient object detector BASNet [42] with manually labelled images (i.e. the boundaries of the phalanges are drawn by a human annotator). This step yields initial segmentation masks as shown in Figure 3a.

3.2.2 Fingertip Detection and Segmentation Refinement. To estimate the fingertip location along the segmentation contour, we first fit a polynomial curve to the mask pixels using regression. There are two possible curves that can fit the pixels. The correct curve is identified as the one that aligns more closely with the orientation of the phalanx (determined from the aspect ratio of the bounding box encapsulating the segmentation contour). The curve intersects at least two points along the contour, one of which is the approximate fingertip position. We determine the correct point, based on the knowledge of which hand is visible. For instance, for the rear view, if the digit is the left thumb, we pick the leftmost intersection point. For the top view, we choose the top point (Figure 4).

After identifying the fingertip on the contour, we refine the phalanx mask by applying a circular filter centred at the fingertip. The filter's radius is determined experimentally to exclude pixels beyond the distal interphalangeal joint, effectively retaining only pixels belonging to the distal phalanx (Figure 3b).

3.3 3D Pose And Location Estimation

The final step of the data-creation pipeline is the determination of the 3D pose of the distal phalanx. To achieve that, we use differentiable mesh rendering, which is a self-supervised computer vision technique that can find correspondences between 3D objects and their appearances in 2D images [24, 25]. In our case, a differentiable renderer optimises the 6-DoF position and orientation of

a 3D phalanx mesh, so that its rendered form best overlaps with the 2D mask. For the mesh, we use the MANO hand model [44], from which we extract submeshes of the distal phalanges of the thumb and the index finger. Using this rigid, non-articulated subset of the finger mesh allows us to optimise only for 6-DoF position and orientation rather than for the complex articulated pose of a full hand. While people's finger sizes and shape differ, this model provides a sufficient approximation for our purpose.

For a considered phalanx mesh, rotation and translation matrices that orient and position the mesh are determined through an optimisation process, which iteratively minimises a loss function through gradient descent (Figure 3c). At each step, the mesh is rendered for each virtual camera using Soft-Rasterizer [32] and the corresponding loss is calculated using three constituent losses: (1) a *Silhouette Loss*, which steers the 3D mesh towards the mask location; (2) a *Distribution Alignment Loss*, which attempts to align the mesh silhouette and mask distributions for faster convergence; and (3) a *Tip Distance Loss*, which tries to align the projected 3D fingertip with the mask's 2D fingertip.

The *Silhouette Loss* is defined as the weighted distance between the projection of MANO's distal phalanx mesh into 2D space using Soft-Rasterizer and the refined BASNet mask obtained in the previous step:

$$L_{\text{silhouette}} = \|\mathbf{w}(\mathbf{m_{proj}} - \mathbf{m_{est}})\|_2^2 \tag{1}$$

where $\mathbf{m_{proj}}$ is the mesh projection, $\mathbf{m_{est}}$ the BASNet mask and $\mathbf{w}$ is a non-linear 2D matrix. Its weight component $w(\mathbf{p})$ for a pixel coordinate $\mathbf{p}$ is defined as:

$$w(\mathbf{p}) = \begin{cases} e^{\frac{-d}{\max d}} & \text{if } d < \theta_{\text{thres}} \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

where θ_{thres} is the circle radius defined in the previous mask refinement step that corresponds to the approximate distal phalanx length, and $d = \|\mathbf{p} - \mathbf{t}\|_2$, with $\mathbf{t}$ being the estimated coordinates of the fingertip location.

The *Distribution Alignment Loss* is defined as the sum of the two normalised differences of pixel histograms of the rendered silhouette and the phalanx mask on each axis:

$$L_{\text{dist}} = \frac{1}{W}||\mathbf{m}^{\mathbf{x}}_{\mathbf{proj}} - \mathbf{m}^{\mathbf{x}}_{\mathbf{est}}||_2^2 + \frac{1}{H}||\mathbf{m}^{\mathbf{y}}_{\mathbf{proj}} - \mathbf{m}^{\mathbf{y}}_{\mathbf{est}}||_2^2 \tag{3}$$

where W and H are respectively the image width and height, and $\mathbf{m}^{\mathbf{x}}$ and $\mathbf{m}^{\mathbf{y}}$ are the sums of the mask's pixels on the X and Y axes respectively.

The *Tip Distance Loss* ensures fingertip alignment and is defined as the distance between the 2D fingertip estimated in the previous step and the projected 3D fingertip:

$$L_{\text{tips}} = \|\mathbf{Tip_{proj}} - \mathbf{Tip_{est}}\|_2^2 \tag{4}$$

where $\mathbf{Tip_{proj}}$ is the projection of the MANO model's tip vertex on the rendered silhouette image and $\mathbf{Tip_{est}}$ is the tip position on the mask.

Finally, we combine these three loss functions to form a *General Loss* for the rear and top virtual camera views:

$$L(\mathbf{R}, \mathbf{T}) = L^{\text{rear}}_{\text{silhouette}} + \alpha L^{\text{rear}}_{\text{dist}} + \beta L^{\text{rear}}_{\text{tips}} + L^{\text{top}}_{\text{silhouette}} + \alpha L^{\text{top}}_{\text{dist}} + \beta L^{\text{top}}_{\text{tips}} \tag{5}$$

where α and β are hyperparameters, and L^{rear} and L^{top} are the loss functions applied to the images of the virtual rear and top cameras respectively. $L(\mathbf{R}, \mathbf{T})$ is minimised to obtain the rotation $\mathbf{R}$ and translation $\mathbf{T}$ of the phalanx mesh, which best matches the finger's appearances in the image pairs. Fig. 5 shows an example of this optimisation process, which results in the 3D phalanx mesh aligned with the corresponding phalanx pixels in the input images.

The 3D pose of the phalanx is estimated for each frame separately, but we can expect the fingertip positions to remain close in consecutive frames so we initialise the values of $\mathbf{R}$ and $\mathbf{T}$ with the values estimated in the previous frame. This decreases the number of iterations required to arrive at an optimal solution.

3.4 End-To-End Model

While the data-creation process gives us an estimate of 3D fingertip positions, it is typically slow and therefore not suitable for real-time inference required for tracking. We therefore use the phalanx poses estimated in the data-creation process as training data for a dedicated end-to-end neural network, which is used for real-time inference. The goal for such a model is to directly output the pose estimate of the distal phalanx as well as a probability value indicating to which finger it belongs, or "no finger" if no interacting fingers appear in the image (fingers used to hold the phone do not count as "interacting" fingers).

We use Ge et al.'s graph CNN-based model as our neural network base [16], which is a popular architecture for 3D hand pose estimation. It expects single-view RGB images as input, so we perform the forward pass of the pre-trained encoder with the rear and top images individually and concatenate the latent features obtained as output. We then feed the concatenated feature maps to the multi-layer perceptron (MLP) branch of the network to obtain the desired 3D pose and finger visibility probabilities (Figure 6).

We use two loss functions L_{coord} and L_{rend} to train the network. The model predicts the rotation $\hat{\mathbf{R}}$ and the translation $\hat{\mathbf{T}}$, which transform the phalanx mesh into a positioned mesh $\hat{\mathbf{V}}$, from which the fingertip position can be obtained. The L_{coord} loss is the Huber distance between $\hat{\mathbf{V}}$ and $\mathbf{V}$, where $\mathbf{V}$ is the ground truth 3D pose of the phalanx obtained by applying $\mathbf{R}$ and $\mathbf{T}$ to the mesh. L_{rend} is a render-and-compare loss that compares the rendered silhouette $\hat{\mathbf{m}}_{\mathbf{proj}}$ with the mask $\mathbf{m_{est}}$ in image space (Eq. 6).

$$L_{\text{rend}} = \|\hat{\mathbf{m}}_{\mathbf{proj}} - \mathbf{m_{est}}\|_2^2, \tag{6}$$

Finally, we apply L_{cls}, a binary cross entropy loss, to determine which interacting finger is visible in the image (if any). The combined loss function L_{verts} is defined as follows:

$$L_{\text{verts}} = \lambda_1 L_{\text{coord}} + \lambda_2 L^{\text{rear}}_{\text{rend}} + \lambda_3 L^{\text{top}}_{\text{rend}} + L_{\text{cls}}, \tag{7}$$

where λ_1, λ_2, and λ_3 are hyperparameters. While L_{cls} is always applied, L_{coord}, $L^{\text{rear}}_{\text{rend}}$, and $L^{\text{top}}_{\text{rend}}$ are only applied when corresponding fingers are visible in the input images.

Using L_{verts}, we fine-tune both the encoder and the MLP components of our neural network to create our fingertip-detecting model. We use standard data augmentation operations such as Gaussian filters, cutout, random pixel intensity adjustments to increase the robustness of the model.

3.4.1 Handling multiple fingers. Although the data-creation pipeline described above is designed to predict the pose of a single fingertip, it can be easily extended to handle multiple fingers without manually collecting additional data using mixup data augmentation [63].

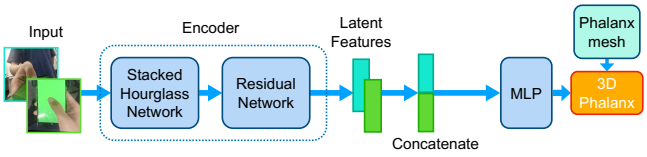


Figure 6: Adaptation of Ge et al.'s hand pose estimation network architecture [16] to handle input from multi-view image pairs. Each view is fed individually to the encoder. The two output latent feature vectors are concatenated in a single vector. This vector is, in turn, fed to the MLP branch that predicts the 3D fingertip pose and finger presence probability.

Specifically, a source image is randomly chosen from the dataset, then a check is performed to determine which finger is shown (if any), and a complementary finger image is randomly picked (i.e. an image of a thumb or index finger from the other hand) to create a pair of images. If no finger is present in the source image, an image that includes any finger is randomly chosen. The two selected images are blended using the formula

$$I_{\text{aug}} = \gamma I_{\text{src}} + (1-\gamma) I_{\text{tar}}, \tag{8}$$

where $\gamma \sim U(0.3, 0.7)$. The mask of the combined image, which is needed for L_{rend}, is determined by performing a Boolean OR operation on the masks of the two source images. $\mathbf{R}$ and $\mathbf{T}$ estimates and finger presence probabilities are extended accordingly to handle multi-fingers and multi-labels in Eq. (7).

Models trained in such a way can be deployed for two-hand input, such as two-thumb typing (Figure 1c).

4 VR CLIENT

Transforming the inferred 3D positions of detected fingertips into a functional VR hand representation has two challenges: 1) To create full hand representations, the hand joints need to be derived from only the fingertip positions and pose constraints when using the phone. 2) To enable precise touch input, touch positions and contact events need to be determined in a way that is consistent with the hand representation and tactile feedback when touching the phone screen.

4.1 Generating Hand Representations

A simple solution for a hand model based on fingertip positions is a single solid 3D object representing the fingertips, such as a 3D cursor (Figure 7d). However, deriving the positions of other parts of the hand to create a full hand representation (e.g. Figure 7a) is more challenging. We make assumptions about how the physical hand is positioned relative to the phone depending on which fingertip is tracked to create full hand models.

To construct a hand pose based on the thumb tip position, we assume only the thumb moves to operate the phone and the rest of the hand remains mostly still. This is a simplification because in reality the user slightly adjusts their grip on the device to reach different areas of the touchscreen, but we hypothesise that attempting to faithfully reflect complex re-grasping hand motions in VR is less important than accurately representing the position and motion of the thumb tip. With the virtual hand having a fixed grip on the phone, the articulated joints of the thumb can be inferred using inverse kinematics from the thumb tip. This is similar to how commercial VR systems render hands gripping controllers using a fixed pose and constrained animations approximating finger motions when pressing buttons.

To create a hand pose from the index fingertip position, we assume the rest of the hand is not gripping the phone. Again, we make a simplification by rendering the hand in a fixed pose with extended index finger (such as shown in Figure 1b). The entire virtual hand moves based on the fingertip position and disappears when the fingertip is not tracked.

4.2 Determining the Touch Point

Due to tracking imperfections, discrepancies between estimated and real fingertip positions might exist, which can affect touch input performance, as the virtual finger may appear slightly above or below the virtual phone screen upon physical contact [67]. To reduce this visual-haptic mismatch, we perform a fine-grained calibration on a 4×7 grid of reference points, where each point is successively tapped and the height offset of the estimated fingertip positions at those locations recorded. The z-value of inferred fingertip positions can then be adjusted using local bilinear interpolation between the closest reference points to more closely match virtual collisions with physical contacts. This calibration is only performed once after training a new inference model, not for each user.

To determine the touch point on the virtual phone screen, we orthogonally project the centre of the fingertip volume (which is at a fixed position from the fingertip point estimated by the tracker) onto the screen surface, similar to the method used by Zhu et al. in their dynamic calibration process[67]. We use this projected point instead of the touch coordinates reported by the phone's capacitive sensor to maintain continuous visual consistency of the touch point based on the virtual hands' finger positions (which may slightly differ from real fingers' positions).

4.3 Enabling System

We create a proof-of-concept implementation of our system using the following hardware and software components.

Mobile Phone. For the phone, we use a Google Pixel 3, which has a 5.5 inch screen and a front camera located on the upper left side. We create a mount for the two mirrors with an articulated arm made of acrylic glass attached to a holder clipped to the phone (Figure 1a). The front-view mirror of size 65×65mm is fixed at a 70° angle just above the front camera and the top-view mirror of size 65×80mm is placed 78mm above the screen. To spatially track the phone with high precision and without attaching a heavy VR tracker to the device, we use an Optitrack motion capture system that detects a constellation of small spherical optical markers attached to the mount. We use this system only to track the position of the phone (and the headset), not to track the user's hand. The entire mount weighs 116g (100g without the markers), which add to the 156g of the Pixel 3.

We create a custom Android application that crops and transforms the two mirror images as described above (Figure 2), then compresses them to jpg before streaming the images to the inference server via WiFi. To achieve high frame rates, we set the phone

to capture at a resolution of 400×640px and send 256×512 images consisting of the two concatenated mirror views. This allows the phone to capture and stream at a constant 30 fps. Touch input events are sent across the same streaming channel.

Neural Network Implementation. To gather training data for our neural network, we ask 11 people in our institution (8 male, 3 female) to hold the phone in each hand successively and move their thumb and index finger on and over all regions of the screen for a few minutes, while the phone records images captured by the front camera. We further collect images where no finger is manipulating the phone for our "no finger" condition. We feed the gathered data to our data creation pipeline, where we manually annotate 350 images for each digit to improve BASNet segmentation. We fine-tune the BASNet model for each finger, which ensures that for the segmentation of index fingers, visible thumbs of the hand holding the phone are not masked instead. We further review the generated pose labels to eliminate visibly incorrect outputs ($\sim 8\%$ of the dataset, with $\sim 2\%$ resulting from incorrect segmentations). After this filtering operation, we obtain $\sim 30,000$ images for the left hand and $\sim 35,000$ images for the right hand, both with an equal amount of images for the thumb and index finger, and $\sim 6,500$ "no finger" images for each hand. Our neural network is implemented in PyTorch with PyTorch3D [43] used for differentiable rendering. The models are trained following Eq. (7) for 25 epochs using the Adam optimiser.

Note that while new data and inference models would have to be created for other phones with different camera positions and mirror configurations, segmentation of distal phalanges through our fine-tuned BASNet model should be sufficiently general that the model can be reused without labelling new images. We have verified this with images captured by two other phones, suggesting our method is entirely self-supervised after this initial step.

Inference Server. The inference server that receives the input frames is a Ubuntu PC with an AMD Ryzen 9 5900X and GeForce RTX 3090 GPU. The server runs a Python program with the tracking algorithm to infer the 3D position of the fingertip of the considered hand configuration. We optimise inference speed with the TensorRT framework and achieve an average inference time of $\sim 5ms$ per frame. We apply temporal filtering on the obtained position vectors using the 1 euro filter [9] to reduce noise and send the resulting vector along with touch input information to the VR client.

VR Client. For technical reasons, the VR client is deployed on a separate Windows PC to which a Vive Pro VR system is connected. The VR applications are developed in Unity and a scene with just the phone and an animated 3D hand model runs consistently at a display frame rate above 200fps.

With a high-speed camera, we measure the end-to-end latency of the entire processing chain, i.e. the time difference between the physical movement of a real finger and when that movement is reflected by the virtual hand in the VR environment. We obtain an average latency of $\sim 125ms$ (SD=11.3).

4.4 3D Tracking Precision Evaluation

To assess the tracking precision of our model, we compare the fingertip positions estimated by our tracker with the 3D positions of an optical marker affixed to the top of the nail, as captured by the OptiTrack system. We recruit ten people, who did not contribute data for training, and ask them to perform tapping motions on the whole screen successively with their thumb and index finger for three minutes. After aligning the coordinate systems and event time sequences of our tracker and those of the Optitrack system, we compute the root mean squared error (RMSE) for each participant and each finger and obtain an average RMSE of $\sim 7mm$ (SD=1.0) for the thumb and $\sim 8mm$ (SD=1.2) for the index finger. These results include some edge cases with fingers high above the screen or near the phone bezel, where visibility is reduced. When only considering fingers touching the screen or directly hovering up to 2cm above it, i.e. the main interaction area, the RMSE becomes $\sim 6mm$ for both fingers (SD=0.7 for the index finger and 0.6 for the thumb). These results are encouraging as our fingertip tracker for hands holding and interacting with phones demonstrates greater precision than barehand trackers of commercial HMDs. [1, 47]. Furthermore, since we use the projection of the inferred fingertip position and not the capacitive sensor of the phone to determine the touch point, small deviations between real and virtual positions may not be perceived by the user and thus may have only a minimal impact on precision. To confirm this hypothesis and determine the suitability of our fingertip tracker for precise touch input, a dedicated user evaluation with a VR task is required.

5 TOUCH INPUT EXPERIMENT

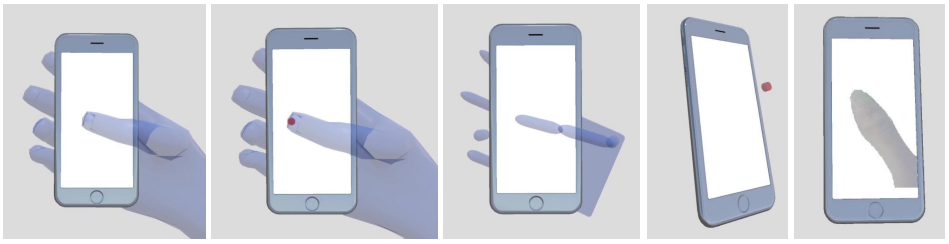
By tracking fingertips in 3D space, our system is capable of supporting a variety of phone-driven VR scenarios involving touch and mid-air interaction (see "Applications" further below), but as a first experimental usability validation, we focus on determining whether tracking is sufficiently robust to reliably control virtual hands for precise single-finger touch input, which is a fundamental requirement for general support of smartphones in VR.

The performance of direct hand interaction in VR not only depends on tracking accuracy, but also on how the hand is represented [5, 17, 27]. As it is unclear which virtual hand shape and appearance can best facilitate precise touch input in our particular context, we seek to compare different styles that may potentially affect touch performance.

Following those considerations, we conduct a controlled experiment based on simple tapping and tracing tasks using the thumb and index finger with different virtual hand representations. Based on obtained results, future investigations can select the most effective hand appearances and focus on other evaluation criteria, tasks and applications.

5.1 Hand Representations

As identified in prior work on hand representation for VR precision tasks [17, 27], there are conflicting design objectives between realistic hand representations that maximise the user's sense of embodiment and presence but can be less precise (because of fat fingers and occlusions) and more abstract hand appearances designed to improve targeting, but with lower fidelity. We explore this tension by creating different hand models along that spectrum. To limit the number of influencing design factors, we only consider geometry as variable and fix other attributes such as texture



(a) NORMAL-SHAPE (b) NORMAL+MARKER (c) STICK-HAND (d) 3D-CURSOR (e) 2D-OVERLAY

Figure 7: VR hand representations compared in the study. a) Hand with normal shape, b) Hand with normal shape with fingertip marker, c) Abstract hand with stick fingers, d) 3D cursor, e) 2D overlay [34].

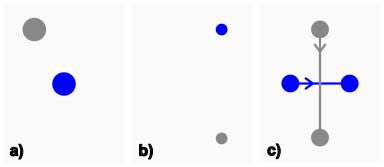


Figure 8: Types of trials used in the experiments (current trial in blue, next trial in gray): a) target acquisition trial with large target, b) target acquisition trial with small target, c) tracing/swiping trial

and transparency. Specifically, we choose a semi-transparent blue texture, similar to hand models used in standard VR toolkits and applications. This ensures that participants do not identify more or less with an arbitrary skin texture or colour and allows the content behind the hand to remain visible through semi-transparency, which can improve accuracy [55].

Figure 7 shows the five VR hand representations that we consider (four 3D representations + Phonetroller's 2D camera overlay), to which we add a NO-VR baseline condition, in which the phone is used normally without VR. Comparisons between VR hands will determine which representations are more suitable for precise touch input, while comparisons between the most effective 3D VR hand with 2D-OVERLAY and NO-VR will reveal how well the tracker performs.

Note that for 2D-OVERLAY, no inference is performed as the camera feed showing the segmented hand is directly displayed on the virtual phone screen and the touch point of the phone's capacitive sensor is used for input. To support this technique, we create a second mount with a single 60×85mm mirror placed 78mm above the phone screen, similar to the system used in Phonetroller [34]. The mount weighs 89g, which is 27g less than the two-mirror mount. Since 2D-OVERLAY directly streams the camera feed to the VR client, the inference server is not used in this case. A further requirement for this condition is to perform a per-user touch calibration as the mirrored view has a skewed perspective compared to normal phone use. We use the procedure described in Phonetroller, i.e. four crosshairs that have to be tapped successively to create the perspective transform which aligns the touch point with the user's intended input.

5.2 Protocol

5.2.1 Tasks. Our experiment is based on Phonetroller's controlled tasks, which consist of simple target acquisition trials. Circle targets are successively shown in a fixed order starting from the centre of the screen and moving around its four corners in a clockwise direction, with a central target tap required between each step. 12 successfully tapped targets are required to complete a round. As in Phonetroller, we use two target sizes, a small size of 5mm, corresponding to the width of a key of the default keyboard and a large size of 10mm, corresponding to typical button sizes like the app icons on iOS (Figure 8a and b). A round of large targets is followed by a round of small targets, for a total of 24 targets. Participants are required to tap inside the circle to validate the trial and proceed to the next one. If a participant fails to hit a target, they must repeat the previous trial (t-1), thereby encouraging accuracy to minimise errors. If the participant misses the t-1 target, however, the sequence does not backtrack further to trial t-2. Instead, the participant continues to retry the same t-1 trial until successful.

In addition to those tapping trials, we introduce tracing trials (which are absent in Phonetroller's evaluation) that require the participant to swipe as precisely as possible along vertically and horizontally centred lines in both directions, i.e. 4 trials (Figure 8c). The tracing error is calculated as the variance of the distances between the touch points of the swiping gesture and the displayed line. To prevent participants from being too careful and slow, we include a time constraint of 2 seconds within which the tracing trial has to be completed.

A full block of tapping and tracing trials therefore consists of 12 tap targets × 2 tap target sizes + 4 tracing trials = 28 trials. We include two blocks, i.e. 2 × 28 = 56 trials, preceded by 5 tapping and 4 tracing warmup trials, which are not logged. For each hand representation or condition, participants perform these 9 warmup + 56 main trials with one digit (thumb or index finger), followed by the other digit, i.e. (9 + 56) × 2 = 130 trials, of which only 112 are logged. The number of trials increases if participants miss tap targets or swipe too slowly.

In total, our experiment consists of a minimum of 56 main trials × 2 digits × 6 conditions = 672 logged trials.

5.2.2 Design. Our study follows a within-subjects design, in which participants perform the tasks with all six hand representations. The orders of the conditions and the start digit for each condition are rotated between participants according to a balanced Latin square to mitigate the influence of learning effects on the results.

Our primary independent variables are REPRESENTATION with six levels (NORMAL, NORMAL+MARKER, STICK-HAND, 3D-CURSOR, 2D-OVERLAY and NO-VR), and TARGET-SIZE with two levels LARGE and SMALL for tapping tasks. Our dependent variables consist of the measured values TAP-TIME and TRACE-TIME, denoting the mean times taken to successfully complete individual tapping and tracing trials, TAP-ERRORS and TRACE-ERROR as defined above, and participants' subjective ratings for their sense of CONTROL and PRESENCE in the post-study questionnaire.

We refer below to NORMAL+MARKER, STICK-HAND and 3D-CURSOR collectively as "precision representations" as their design is optimised for precise touch input, compared to NORMAL and 2D-OVERLAY which have no such optimisations.

5.2.3 Participants. We recruit 18 participants from our institution, 5 female and 13 male with average age 36.7 years (SD=8.5). Four of those participants contributed data for the training of the neural network. Regarding VR experience, three people have never used a VR system before, thirteen experienced VR one to three times, and two have previously used VR systems on a monthly basis for games and participation in virtual events.

5.2.4 Procedure. The experiments are performed in a sitting position with the apparatus described above. Participants can adjust the chair and armrests to their preference and are permitted to rest their elbows on the armrests during the task to reduce fatigue.

Participants start each condition by performing training tasks with the required digit. For thumb input, the phone is to be held in the dominant hand and the thumb of the same hand used to perform the tasks. Participants are given permission to lightly support the device with their other hand if they wish. For input with the index finger, the phone should be held in the other hand and the index finger of the dominant hand used to perform the tasks. When participants feel they have sufficiently trained, they proceed to the main task consisting of the warmup and logged trials. After completing all trials, they repeat the same procedure with the other digit. This process is in turn repeated for the remaining five conditions.

For 2D-OVERLAY, the two-mirror mount is replaced with the single-mirror mount and the calibration procedure described above is performed before starting the tasks. We allow participants to recalibrate for the second digit if they feel that the initial calibration for the first digit does not work. This calibration step is not performed for the 3D VR hand representations and NO-VR.

After completing all six conditions, participants are asked to answer a questionnaire about their experience. Since the study has six conditions, we limit our questions on participants' subjective impressions to the two general aspects of control and presence. Specifically, we ask people to rate their sense of control (how well they perceived to have been in control of the virtual hands) for each condition and sense of presence (how much they felt present and embodied in VR through the hands) for each VR condition on a linear scale from 1 (lowest) to 10 (highest). Next, we ask them to choose their preferred representation overall for precise touch tasks. Finally, a free comment section allows participants to provide feedback on any aspect of the experiment.

Completing the questionnaire concludes the study after roughly one hour and people are offered a choice of snacks to thank them for their participation.

5.3 Results

5.3.1 Quantitative Results. We first consider the possible influence of including participants who provided data to train the tracker as well as learning effects between trial blocks. Respectively independent and dependent T-tests on mean times and errors for each trial type are not significant in both cases (all p values>0.2), suggesting that neither factor has an influence on the results. We therefore include all logged data for our analyses, the results of which are summarised in Figure 9.

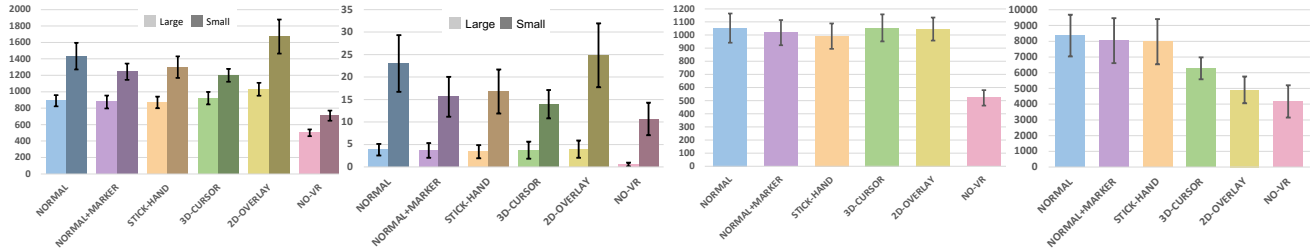
To analyse our data, we perform repeated measures ANOVAs, confirming distribution normality in each case and applying Greenhouse-Geisser corrections when the sphericity condition is violated.

We first consider tap trials. We perform two-way ANOVAs for REPRESENTATION ×TARGET-SIZE on TAP-TIME and TAP-ERRORS and find significant interaction effects in both cases ($F(3.3, 56.5) = 10.8$ and $F(5, 85) = 5.16$, both $p < 0.001$), so we analyse the data for each target size separately. For large targets, we obtain significant effects for both TAP-TIME ($F(5) = 94.5, p < 0.0001$) and TAP-ERRORS ($F(5) = 4.01, p = 0.002$) so we conduct post-hoc tests with Holm–Bonferroni corrections. We find that all VR conditions are significantly slower and more error-prone than NO-VR (all $p < 0.035$). Additionally, 2D-OVERLAY is significantly slower than all other techniques (all $p < 0.012$). Specifically, participants were able to correctly tap large targets in 501ms on average (SD=89) with NO-VR, which is more than half the average time of 2D-OVERLAY (1029ms, SD=170) and 56 58% of other VR conditions.

For small targets, we obtain again significant effects for both TAP-TIME ($F(3.2) = 38.5, p < 0.0001$) and TAP-ERRORS ($F(5) = 6, p < 0.0001$) so we perform pairwise comparisons. These tests reveal that all VR conditions are significantly slower than NO-VR (all $p < 0.0001$). Furthermore, all precision representations are significantly faster than 2D-OVERLAY ($p < 0.005$) and 3D-CURSOR is significantly faster than NORMAL ($p = 0.023$). Concretely, the slowest technique, 2D-OVERLAY, exhibits a tapping time of 1671ms (SD=450), which is more than 2.3 times the tapping time of NO-VR (709ms, SD=133) and approximately 1.4 times the tapping times of the precision representations (1200ms, SD=172 for 3D-CURSOR and 1243ms, SD=215 for NORMAL+MARKER). With regard to errors, only 2D-OVERLAY and NORMAL are significantly more inaccurate than NO-VR ($p < 0.011$). The average number of errors for NO-VR (10.7, SD=7.8) is only slightly lower than for precision representations (14, SD=60.8 for 3D-CURSOR and 15.6, SD=9.6 for NORMAL+MARKER), with differences not statistically significant.

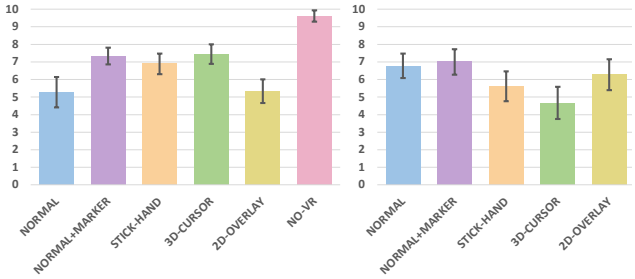
Altogether, these results show that for tapping tasks, the precision representations outperform 2D-OVERLAY in terms of speed and accuracy.

Turning to tracing trials, we perform ANOVAS on TRACE-TIME and TRACE-ERRORS and obtain significant effects in both cases ($F(5) = 80.1, p < 0.0001$ and $F(5) = 10.33, p \ll 0.0001$) so we perform post hoc tests. For TRACE-TIME, we find that NO-VR is significantly faster at 521ms (SD=128) than all VR conditions ($p < 0.0001$), which average 1019-1055ms. NO-VR, with a mean trace error of 4178ms (SD=2225), is also significantly more precise than all 3D VR representations ($p < 0.013$), where 3D-CURSOR averages at 6281 (SD=1505) and the three other representations exhibiting errors around 8000. The error difference between NO-VR and 2D-OVERLAY (4908, SD=1850) is not statistically significant (p=1). 2D-OVERLAY, in turn, exhibits a significantly lower error compared to all 3D VR conditions (all $p < 0.022$) but not 3D-CURSOR ($p = 0.32$). All other differences are not significant. We believe the higher tracing precision of NO-VR and 2D-OVERLAY compared to almost all 3D VR conditions can be attributed to the fact that the former techniques use the phone's touch sensor for the touch point, which seems more stable. Despite the temporal filtering, it appears the inferred 3D



(a) Mean tap trial time (ms) (b) Number of tap errors (c) Mean trace trial time (d) Aggregate trace error

Figure 9: Quantitative results of the experiments. Error bars represent 95% confidence intervals.



(a) Sense of Control (b) Sense of Presence

Figure 10: Participant ratings of their sense of control for all conditions and their sense of presence for all VR representations on a continuous scale from 1=worst to 10=best. Error bars represent 95% confidence intervals.

fingertips still suffer from light noise, whose impact is amplified by an error calculated on squared distances.

5.3.2 Participant Ratings. We examine participants' ratings of their sense of CONTROL for all conditions and their sense of PRESENCE for the VR representations (Figure 10). An ANOVA for CONTROL shows significant effects ($F(3.03) = 25.91, p < 0.001$), so we perform pairwise comparisons. The tests show that participants felt significantly more in control with NO-VR compared to all VR conditions (all $p < 0.001$). They furthermore rated control significantly lower for 2D-OVERLAY and NORMAL compared to the precision representations (all $p < 0.044$).

For PRESENCE, an ANOVA reveals a significant effect ($F(2.85) = 7.18, p \ll 0.001$). Post hoc tests show that 3D-CURSOR is rated as having a significantly lower presence than NORMAL and NORMAL+MARKER (both $p < 0.002$). Differences in all other comparisons are not significant.

Overall, 8 participants selected NORMAL+MARKER as their preferred hand representation, followed by 3D-CURSOR (5 participants), STICK-HAND (4 participants) and 2D-OVERLAY (1 participant). These choices and ratings along with the measured results confirm the general superiority of the precision representations over the two other VR conditions, with a slight caveat for tracing stability.

Examining participants' comments, we find that people who preferred NORMAL+MARKER valued the combination of a familiar hand shape for high presence and a marker on the fingertip for high precision, essentially a combination of NORMAL and 3D-CURSOR. Participants who preferred 3D-CURSOR, however, appreciated its minimalist appearance, which for them provided more control because they were not distracted by a virtual hand model, whose pose did not always match with that of their real hand. For three participants, these real-virtual mismatches disrupted proprioception for thumb movements, as the virtual hand and the articulation of its thumb conveyed a false image of possible movements. Participants who preferred the stick hand appreciated the simple design to facilitate aiming without a marker while maintaining recognisable hand features. Two participants, however, stated that using that model felt a bit "strange". The participant who favoured 2D-OVERLAY mentioned seeing their actual hand on the virtual phone screen as the reason for their choice. The direct visual feedback of the user's actual hand that 2D-OVERLAY provides is also the reason why it enjoys a relatively high PRESENCE rating, despite showing only a small portion of the hand and lacking depth.

With regard to perception of detection accuracy for the 3D VR hand representations, three participants noted a degradation of tracking performance when using their index finger while extending other fingers, especially when their index finger was bent. While our training data included images of such cases, it appears our tracker was still sometimes confused by other visible fingers that also seemed poised to touch the screen. When we noticed this reduction in tracking performance we suggested the participant to extend only their index finger, which improved detection.

Another limitation participants noticed for tapping trials is the higher difficulty of hitting targets on the hand side of the phone with their thumb in VR. Paired T-tests for each representation comparing the tap errors for left targets and right targets confirm statistically significant differences for 3D-CURSOR ($p < 0.001$) and HAND+MARKER ($p = 0.039$), while 2D-OVERLAY is borderline ($p = 0.055$). We attribute this lower performance to the reduced visibility of the hand and flexed thumbs on the phone side, which we also observed in the 3D tracking precision evaluation.

6 DISCUSSION

Our experiments showed that our VR system cannot yet fully replicate the touch input experience and performance of real-world phone use. Beside the precision of fingertip estimation, we believe factors, such as the additional weight of the mount (as mentioned by

some participants) and latency[2, 26, 27, 34], contribute to that diminished experience. Furthermore, our trials demanded significant finger movement and reach for the thumb when tapping from one corner of the screen to another. We believe speed, tapping accuracy and tracking stability would be higher if successive targets were close and located in a central region of the screen, e.g. when typing on a keyboard.

We did not compare our VR hand models to 3D augmented virtuality methods such as Bai et al [5] as they require a depth camera on the headset to capture a 3D visualisation of the hand, but we note that those approaches are unable to match the performance of real-world precise phone interaction either, due to various technical limitations. Compared to Phonetroller [34], we showed that using precision representations for 3D hand models increased tapping accuracy. Furthermore, unlike Phonetroller, our system does not require user-specific touch calibration. A possible improvement of Phonetroller's 2D hand shadow would be to also show a marker on the fingertip (the paper proposed a deep learning technique to estimate those "hover points", but it was not evaluated) and to assess how the lack of depth may affect the different performance and perception metrics. However, if fingertip position estimation is not sufficiently consistent, using those keypoints to determine touch points may result in slightly more uneven strokes when dragging fingers on the screen, as shown by the results of our tracing task. More aggressive temporal filtering might mitigate that effect, but possibly at the cost of additional latency.

A few participants commented that not faithfully reproducing the full hand pose and movement in VR was an issue. Contrary to simplifying assumptions to render fixed hand poses, our results suggest even slight differences between virtual and real hands can affect a user's mental model of achievable finger movements. Especially for constrained movements of the thumb of the hand holding the phone, real-virtual pose mismatches at the finger joint level seem to have a higher impact than freehand movements like touching with the index finger of the other hand or typing on a physical keyboard. Those factors need to be taken into account when creating virtual hand models for single-hand phone interaction in VR.

Regarding the impact of different hand representations compared to previous work, we only partially confirm Grubert et al's results [17], which showed that fingertips and a video inlay reduced errors when typing on a physical keyboard. In our case, the 2D camera overlay performed worse for tapping, likely because our targets are smaller and have no haptic feedback like keys. In terms of presence, we confirm Knierim et al's findings that showing only fingertips decreases the sense of presence [27]. Our results align with their recommendation to use realistic hand models to maximise performance and presence, echoing other suggestions in the same direction [22, 48, 60]. However, our results also reveal additional challenges for precise touch input with a phone.

The above considerations lead us to formulate the following recommendations for the design of virtual hand representations for precise touch input on mobile phones in VR: If only the fingertips can be tracked, we suggest only showing simple cursors or pointers, which have the highest precision. That comes at the cost of a slightly lower sense of presence, but it avoids distractions and potential mismatches between real and virtual hand poses and movements. If the full hand pose can be reliably tracked and faithfully rendered in VR, we recommend showing semi-transparent hands with small markers on the fingertips, which do not seem to affect presence or distract people. Adding fingertip markers to hand models may also be beneficial for precise finger interaction with hands tracked by HMD sensors. Design decisions may of course vary based on the considered tasks and application scenarios.

6.1 Limitations

In addition to latency, our system and experimental investigation have other limitations. First, our tracker currently only detects thumb and index fingertips. While it theoretically also supports multitouch and two-hand interaction, we have only evaluated single-finger touch input, and only in a sitting position. We hypothesise that our findings and recommendations likely also generalise to phones being used while standing and with two hands/fingers, however other issues might arise in those contexts that can only be uncovered through experiments specifically addressing those scenarios.

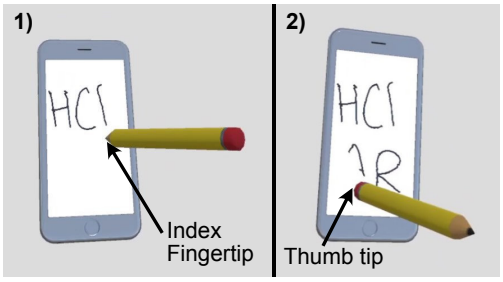
Our deep neural network currently requires a server with GPU to achieve low inference times and we have not tested our method on fully mobile VR systems. However, considering that standalone VR headsets such at the Oculus Quest can track hands in real-time using deep learning [19], we are confident that this is achievable. Our study also used a motion capture system to track the phone, which is not practical for most VR application contexts. Phonetroller attached a conventional VR tracker to the mount, but this adds significant weight to the phone. Lighter 6 DoF tracking solutions could rely on visual odometry using the rear camera and inertial motion sensors, which several mobile AR toolkits support out of the box, albeit with less precision [4, 15].

7 APPLICATIONS

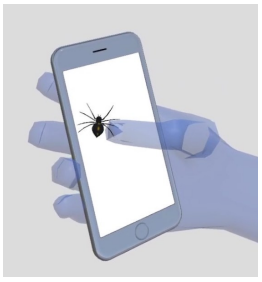
In addition to providing feedback for touch input, tracking fingers in 3D space enables a range of above-screen and touch-to-air techniques, a space explored by prior work in non-VR contexts [12, 21, 61]. In extended reality, hands can interact with virtual objects appearing above the device [13, 39, 51, 66]. We present examples of applications and interactions leveraging our mobile fingertip tracker that go beyond simply replicating physical phone manipulations in VR. We leave the deeper investigation and evaluation of these techniques and the exploration of the underlying design space to future work.

7.1 Object Control with Finger-Differentiated Actions

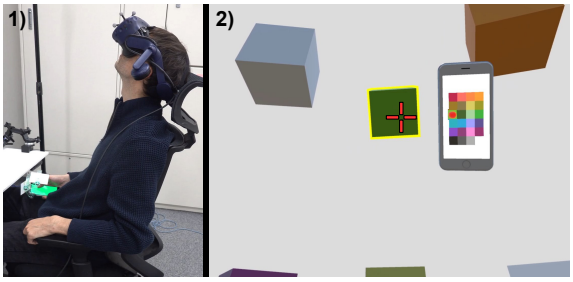
As shown earlier, the tracked fingertip can control a simple object such as a 3D cursor for precise touch input. The fingertip can also be mapped to an object that more closely represents an input instrument for a particular task, similar to how VR controllers can transform into different handheld tools in VR applications (e.g. a pistol or a sword in a game). Task-specific representations and control mappings can be considered at the finger(tip) level as well [54]. For instance, in a sketching application, the index fingertip can control the 3D model of a pen, which virtually inks the screen of the phone (Figure 11aa). Pen control and actions can be swapped based



(a) Sketching application with the fingertip represented as a pen. 1) If the index finger is used, the pen sketches; 2) If the thumb is used, the pen flips to its eraser end and content is deleted.

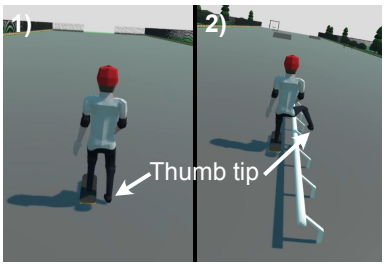


(b) Precise tracking of the 3D thumb tip allows interaction with small virtual objects above the phone model, such as brushing a spider away with the thumb.



(c) Phone UI as head-up display to select texture colours for objects. 1) The user can hold the phone in a comfortable position while looking at virtual objects around and above them without affecting fingertip tracking performance; 2) Objects are selected with head orientation (using a crosshair in the middle of the viewport) and texture colours to assign are chosen by tapping a palette on the phone.

(d) Double raycasting with phone + fingertip to create a lens for map exploration: phone ray sets the centre of the lens, second ray originating from the bottom bezel and passing through the fingertip sets the width and height of the lens.



(e) Skateboarding game scenario, where the thumbtip controls the skater's leg. Dragging on the phone screen causes the skater to push the ground with the foot to move forward. Acrobatic leg movements, such as lifting the leg above a rail, can be performed by moving the thumb above the screen.

(f) Canoeing game example, where the thumbtip controls the oar of a character in a canoe. Dragging on the phone screen rows the canoe. The oar can be lifted to hit overhead targets by raising and moving the thumb above the phone screen.

Figure 11: Example phone-based VR applications utilising our fingertip tracker

on the used finger. For example, when using the thumb instead of the index finger, the pen flips to its eraser end and dragging the thumb on the screen erases content (Figure 11ab).

7.2 Interacting with Above-Screen Objects

The ability to precisely track fingertips above the phone screen enables interaction with small 3D objects in that space. This can be exploited in game scenarios, for example, to brush away spiders crawling on the device with the finger (Figure 11b). Touch input can further be used as a haptic collision event to trigger a different action, such as crushing a spider with the thumb.

7.3 Use as Head-Up Display

Head-up interfaces, which have a fixed position in the 2D screen space of the viewport, are often used in XR to show information to the user. Head-up displays or HUDs can also support phone-operated menus. In such a setting, the phone does not need to be tracked as it is used only for its touch capabilities (with 3D visual feedback of the hand to enable precise targeting). Since fingers are tracked inside-out via the phone itself and not by the HMD, the user is free to look in any direction without impacting fingertip detection accuracy (Figure 11ca). This also allows the phone to be held in a comfortable, low-fatigue position, such as resting on the user's lap or a table when sitting. This scenario cannot be supported by existing VR systems with sensors integrated in the HMD, which cannot track hands that are occluded or out of capturing range. We present an application of this HUD concept, where head orientation or gaze is used to point at 3D objects located around the user in the VR space and the phone is used to choose texture colours for selected objects from a palette (Figure 11cb).

7.4 Double Raycasting

The tracked phone can be used as a 6-DoF raycasting source in the virtual 3D space like a standard VR controller [4, 34, 46]. Since the fingertip is also precisely tracked in the local 3D space of the phone,

it can become an anchor for a second ray to support double raycasting with a single hand, similar to barehand pointing techniques using multiple fingers for remote interaction with large displays [6, 36]. Our example in VR considers a resizable rectangular lens or filter to explore maps, such as revealing the satellite image of a specific portion of the map (Figure 11d). The centre of the lens is determined by the phone ray, and the width and height by a second ray emanating from the centre of the bottom bezel of the phone and passing through the fingertip. If needed, an action (e.g. a selection confirmation) can be triggered by pressing the physical phone volume button on the side of the device.

7.5 Touch/Drag Navigation + Mid-Air Interaction

Scenarios in which the representation of and interaction with the physical phone are significantly abstracted in the VR world can also be considered. Expanding on the idea of using both touch and above-screen input, we propose a novel interaction paradigm for games, in which the tracked fingertip is mapped to the control point of a character and dragging on the phone screen moves the character. We present two game scenarios in which dragging on a surface is a strong metaphor for motion control: skateboarding and rowing. In the first case, the fingertip is mapped to the foot of a skateboarder (Figure 11e): Dragging on the phone screen pushes off the ground to move the board forward, with the foot and leg motion of the skateboarder following the physical movements of the user's finger on and above the touchscreen. In the rowing scenario, the fingertip is mapped to the oar of a character sitting in a canoe (Figure 11f) and dragging the finger on the screen causes the character to row the canoe. The dragging length, speed and direction control the speed and direction of the boat via the virtual rowing actions.

Mid-air interaction can also be an integral part of the game, for example the skateboarder can perform acrobatic moves with the leg (Figure 11e2) and the rower can raise the oar to try to hit overhead targets when passing under them (Figure 11f2). The orientation of the phone as detected by its internal motion sensors can further be used to tilt the skateboard/canoe for additional control. Here again, the phone does not need to be tracked, making these game scenarios easily deployable on inexpensive mobile 3-DoF VR systems which do not track any devices or hands. Furthermore, we speculate that single-finger control of a character with a phone that can be held in a relaxed pose is likely significantly less fatiguing than using a standard VR controller, whose movements are mapped 1-to-1 to physical leg and arm motion. Dragging with a finger on a phone screen to skate or row is an inexpensive and serviceable approximation of haptic feedback and friction when a real foot pushes off the ground or an oar pushes water backwards. Future evaluations are required to confirm those hypotheses as well as examine potential tradeoffs in terms of sense of embodiment and control.

8 CONCLUSION

We proposed a deep learning technique to track the 3D position of thumb and index fingertips for precise touch input on a mobile phone in VR using images captured by the front camera reflected through two mirrors mounted on the device. Our method requires no external sensor for training or inference and only a few hundred images need to be initially labelled, with ground truth data then mainly generated in a self-supervised manner through differentiable rendering. A preliminary evaluation of 3D tracking precision revealed that our technique achieved a mean precision of 6mm in the main interaction area directly above the phone screen. We then used our tracker to control several VR hand models, including "precision hand representations" with markers on the fingertips of a normal hand, an abstract model with thin fingers and a 3D cursor. Our evaluation shows that those precision representations outperform Phonetroller's 2D overlay of the mirrored camera feed for target tapping. Based on those results, we recommended using either 3D cursors or realistic hands with fingertip markers to help users aim more accurately. The main experimental evaluation in this work focused on hand visual feedback for precise touch input, but tracking fingers in 3D enables other types of interaction like 3D object control and above-screen or mid-air input. We presented a few examples of such application scenarios and we plan to investigate that potential more deeply in future work.

REFERENCES

[1] Diar Abdlkarim, Massimiliano Di Luca, Poppy Aves, Sang-Hoon Yeo, R Chris Miall, Peter Holland, and Joseph M Galea. 2022. A Methodological Framework to Assess the Accuracy of Virtual Reality Hand-Tracking Systems: A case study with the Oculus Quest 2. *bioRxiv* (2022).

[2] Karan Ahuja, Vivian Shen, Cathy Mengying Fang, Nathan Riopelle, Andy Kong, and Chris Harrison. 2022. ControllerPose: Inside-Out Body Capture with VR Controller Cameras. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 108, 13 pages. https://doi.org/10.1145/3491102.3502105

[3] Ferran Argelaguet, Ludovic Hoyet, Michael Trico, and Anatole Lecuyer. 2016. The role of interaction in virtual embodiment: Effects of the virtual hand representation. In *2016 IEEE Virtual Reality (VR)*. 3–10. https://doi.org/10.1109/VR.2016.7504682

[4] Teo Babic, Harald Reiterer, and Michael Haller. 2018. Pocket6: A 6DoF Controller Based On A Simple Smartphone Application. In *Proceedings of the Symposium on Spatial User Interaction* (Berlin, Germany) *(SUI '18)*. Association for Computing Machinery, New York, NY, USA, 2–10. https://doi.org/10.1145/3267782.3267785

[5] Huidong Bai, Li Zhang, Jing Yang, and Mark Billinghurst. 2021. Bringing full-featured mobile phone interaction into virtual reality. *Computers & Graphics* 97 (2021), 42–53. https://doi.org/10.1016/j.cag.2021.04.004

[6] Amartya Banerjee, Jesse Burstyn, Audrey Girouard, and Roel Vertegaal. 2012. MultiPoint: Comparing Laser and Manual Pointing as Remote Input in Large Display Interactions. *Int. J. Hum.-Comput. Stud.* 70, 10 (oct 2012), 690–702. https://doi.org/10.1016/j.ijhcs.2012.05.009

[7] 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

[8] Samarth Brahmbhatt, Chengcheng Tang, Christopher D Twigg, Charles C Kemp, and James Hays. 2020. ContactPose: A Dataset of Grasps with Object Contact and Hand Pose. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer International Publishing, Cham, 361–378.

[9] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1 € Filter: A Simple Speed-Based Low-Pass Filter for Noisy Input in Interactive Systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 2527–2530. https://doi.org/10.1145/2207676.2208639

[10] Liangjian Chen, Shih-Yao Lin, Yusheng Xie, Yen-Yu Lin, and Xiaohui Xie. 2021. MVHM: A Large-Scale Multi-View Hand Mesh Benchmark for Accurate 3D Hand Pose Estimation. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 836–845. https://doi.org/10.1109/WACV48630.2021.00088

[11] 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

[12] 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

[13] Bruno R De Araújo, Géry Casiez, Joaquim A Jorge, and Martin Hachet. 2013. Mockup Builder: 3D modeling on and above the surface. *Computers & Graphics* 37, 3 (2013), 165–178. https://doi.org/10.1016/j.cag.2012.12.005

[14] Li Du, ChunChen Liu, Adrian Tang, Yan Zhang, Yilei Li, Kye Cheung, and Mau-Chung Frank Chang. 2016. Airtouch: A novel single layer 3D touch sensing system for human/mobile devices interactions. In *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–6. https://doi.org/10.1145/2897937.2901902

[15] Wei Fang, Lianyu Zheng, Huanjun Deng, and Hongbo Zhang. 2017. Real-Time Motion Tracking for Mobile Augmented/Virtual Reality Using Adaptive Visual-Inertial Fusion. *Sensors* 17, 5 (2017). https://doi.org/10.3390/s17051037

[16] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 2019. 3D Hand Shape and Pose Estimation from a Single RGB Image. In *CVPR*.

[17] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. 2018. Effects of Hand Representations for Typing in Virtual Reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 151–158. https://doi.org/10.1109/VR.2018.8446250

[18] Ewa Gustafsson, Pieter Coenen, Amity Campbell, and Leon Straker. 2018. Texting with touchscreen and keypad phones - A comparison of thumb kinematics, upper limb muscle activity, exertion, discomfort, and performance. *Applied Ergonomics* 70 (2018), 232–239. https://doi.org/10.1016/j.apergo.2018.03.003

[19] 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 (jul 2020), 13 pages. https://doi.org/10.1145/3386569.3392452

[20] Yana Hasson, Gul Varol, Dimitrios Tzionas, Igor Kalevatykh, Michael J. Black, Ivan Laptev, and Cordelia Schmid. 2019. Learning Joint Reconstruction of Hands and Manipulated Objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[21] 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

[22] Sungchul Jung, Gerd Bruder, Pamela J. Wisniewski, Christian Sandor, and Charles E. Hughes. 2018. Over My Hand: Using a Personalized Hand in VR to Improve Object Size Estimation, Body Ownership, and Presence. In *Proceedings of the Symposium on Spatial User Interaction* (Berlin, Germany) *(SUI '18)*. Association for Computing Machinery, New York, NY, USA, 60–68. https://doi.org/10.1145/3267782.3267920

[23] 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* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 528, 15 pages. https://doi.org/10.1145/3544548.3580677

[24] Hiroharu Kato, Deniz Beker, Mihai Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon. 2020. Differentiable Rendering: A Survey. *CoRR* abs/2006.12057 (2020). arXiv:2006.12057 https://arxiv.org/abs/2006.12057

[25] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. 2018. Neural 3D Mesh Renderer. In *CVPR*.

[26] Youngwon R. Kim and Gerard J. Kim. 2017. HoVR-Type: Smartphone as a typing interface in VR using hovering. In *2017 IEEE International Conference on Consumer Electronics (ICCE)*. 200–203. https://doi.org/10.1109/ICCE.2017.7889285

[27] 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

[28] Taein Kwon, Bugra Tekin, Jan Stühmer, Federica Bogo, and Marc Pollefeys. 2021. H2O: Two Hands Manipulating Objects for First Person Interaction Recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 10138–10148.

[29] Khanh-Duy Le, Kening Zhu, and Morten Fjeld. 2017. Mirrortablet: Exploring a Low-Cost Mobile System for Capturing Unmediated Hand Gestures in Remote Collaboration. In *Proceedings of the 16th International Conference on Mobile and Ubiquitous Multimedia* (Stuttgart, Germany) *(MUM '17)*. Association for Computing Machinery, New York, NY, USA, 79–89. https://doi.org/10.1145/3152832.3152838

[30] 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

[31] 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

[32] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. 2019. Soft Rasterizer: A Differentiable Renderer for Image-Based 3D Reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

[33] Brandon J. Matthews, Bruce H. Thomas, Stewart Von Itzstein, and Ross T. Smith. 2019. Remapped Physical-Virtual Interfaces with Bimanual Haptic Retargeting. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 19–27. https://doi.org/10.1109/VR.2019.8797974

[34] 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* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 129, 13 pages. https://doi.org/10.1145/3411764.3445583

[35] Fabrice Matulic, Taiga Kashima, Deniz Beker, Daichi Suzuo, Hiroshi Fujiwara, and Daniel Vogel. 2023. Above-Screen Fingertip Tracking with a Phone in Virtual Reality. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems* (<conf-loc>, <city>Hamburg</city>, <country>Germany</country>, </conf-loc>) *(CHI EA '23)*. Association for Computing Machinery, New York, NY, USA, Article 18, 7 pages. https://doi.org/10.1145/3544549.3585728

[36] Fabrice Matulic and Daniel Vogel. 2018. Multiray: Multi-Finger Raycasting for Large Displays. 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, 1–13. https://doi.org/10.1145/3173574.3173819

[37] Fabrice Matulic and Daniel Vogel. 2022. Terrain Modelling with a Pen & Touch Tablet and Mid-Air Gestures in Virtual Reality. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI EA '22)*. Association for Computing Machinery, New York, NY, USA, Article 301, 7 pages. https://doi.org/10.1145/3491101.3519867

[38] 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

[39] Daniel Mendes, Fernando Fonseca, Bruno Araùjo, Alfredo Ferreira, and Joaquim Jorge. 2014. Mid-air interactions above stereoscopic interactive tables. In *2014 IEEE Symposium on 3D User Interfaces (3DUI)*. 3–10. https://doi.org/10.1109/3DUI.2014.6798833

[40] 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

[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] Xuebin Qin, Zichen Zhang, Chenyang Huang, Chao Gao, Masood Dehghan, and Martin Jagersand. 2019. BASNet: Boundary-Aware Salient Object Detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[43] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. 2020. Accelerating 3D Deep Learning with PyTorch3D. *arXiv:2007.08501* (2020).

[44] Javier Romero, Dimitrios Tzionas, and Michael J. Black. 2017. Embodied Hands: Modeling and Capturing Hands and Bodies Together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)* 36, 6 (Nov. 2017).

[45] Eun Jeong Ryu, Minhyeok Kim, Joowoo Lee, Soomin Kim, Jiyoung Hong, Jieun Lee, Minhaeng Cho, and Jinhae Choi. 2016. Designing Smartphone Keyboard for Elderly Users. In *HCI International 2016 – Posters' Extended Abstracts*, Constantine Stephanidis (Ed.). Springer International Publishing, Cham, 439–444.

[46] Elaheh Samimi and Robert J. Teather. 2022. Multi-Touch Smartphone-Based Progressive Refinement VR Selection. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. 582–583. https://doi.org/10.1109/VRW55335.2022.00142

[47] Daniel Schneider, Verena Biener, Alexander Otte, Travis Gesslein, Philipp Gagel, Cuauhtli Campos, Klen Čopič Pucihar, Matjazz Kljun, Eyal Ofek, Michel Pahud, Per Ola Kristensson, and Jens Grubert. 2021. Accuracy Evaluation of Touch Tasks in Commodity Virtual and Augmented Reality Head-Mounted Displays. In *Symposium on Spatial User Interaction* (Virtual Event, USA) *(SUI '21)*. Association for Computing Machinery, New York, NY, USA, Article 7, 11 pages. https://doi.org/10.1145/3485279.3485283

[48] Valentin Schwind, Pascal Knierim, Cagri Tasci, Patrick Franczak, Nico Haas, and Niels Henze. 2017. "These Are Not My Hands!": Effect of Gender on the Perception of Avatar Hands in Virtual Reality. 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, 1577–1582. https://doi.org/10.1145/3025453.3025602

[49] 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

[50] 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

[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] Xiao Tang, Xiaowei Hu, Chi-Wing Fu, and Daniel Cohen-Or. 2020. GrabAR: Occlusion-Aware Grabbing Virtual Objects in AR. 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, 697–708. https://doi.org/10.1145/3379337.3415835

[53] Catherine Taylor, Murray Evans, Eleanor Crellin, Martin Parsons, and Darren Cosker. 2021. Ego-Interaction: Visual Hand-Object Pose Correction for VR Experiences. In *Motion, Interaction and Games* (Virtual Event, Switzerland) *(MIG '21)*. Association for Computing Machinery, New York, NY, USA, Article 1, 8 pages. https://doi.org/10.1145/3487983.3488290

[54] Wen-Jie Tseng, Samuel Huron, Eric Lecolinet, and Jan Gugenheimer. 2021. FingerMapper: Enabling Arm Interaction in Confined Spaces for Virtual Reality through Finger Mappings. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI EA '21)*. Association for Computing Machinery, New York, NY, USA, Article 194, 4 pages. https://doi.org/10.1145/3411763.3451573

[55] Michiel Van Veldhuizen and Xubo Yang. 2021. The Effect of Semi-Transparent and Interpenetrable Hands on Object Manipulation in Virtual Reality. *IEEE Access* 9 (2021), 17572–17583. https://doi.org/10.1109/ACCESS.2021.3050881

[56] 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

[57] 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

[58] Min-Yu Wu, Pai-Wen Ting, Ya-Hui Tang, En-Te Chou, and Li-Chen Fu. 2020. Hand pose estimation in object-interaction based on deep learning for virtual reality applications. *Journal of Visual Communication and Image Representation* 70 (2020), 102802. https://doi.org/10.1016/j.jvcir.2020.102802

[59] Xing-Dong Yang, Khalad Hasan, Neil Bruce, and Pourang Irani. 2013. SurroundSee: 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

[60] Boram Yoon, Hyung-il Kim, Seo Young Oh, and Woontack Woo. 2020. Evaluating Remote Virtual Hands Models on Social Presence in Hand-based 3D Remote Collaboration. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 520–532. https://doi.org/10.1109/ISMAR50242.2020.00080

[61] 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

[62] 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

[63] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond Empirical Risk Minimization. *International Conference on Learning Representations* (2018). https://openreview.net/forum?id=r1Ddp1-Rb

[64] Li Zhang, Weiping He, Huidong Bai, Qianyuan Zou, Shuxia Wang, and Mark Billinghurst. 2022. A hybrid 2D–3D tangible interface combining a smartphone and controller for virtual reality. *Virtual Reality* (2022), 1–19.

[65] Chen Zhao, Ke-Yu Chen, Md Tanvir Islam Aumi, Shwetak Patel, and Matthew S. Reynolds. 2014. SideSwipe: Detecting in-Air Gestures around Mobile Devices Using Actual GSM Signal *(UIST '14)*. Association for Computing Machinery, New York, NY, USA, 527–534. https://doi.org/10.1145/2642918.2647380

[66] 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

[67] Fengyuan Zhu, Zhuoyue Lyu, Mauricio Sousa, and Tovi Grossman. 2022. Touching The Droid: Understanding and Improving Touch Precision With Mobile Devices in Virtual Reality. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 807–816. https://doi.org/10.1109/ISMAR55827.2022.00099