3D Hand Pose Estimation on Conventional Capacitive Touchscreens

Frederick Choi
Carnegie Mellon University
Pittsburgh, PA, USA
University of Illinois
Urbana, IL, USA
fc20@illinois.edu

Sven Mayer
LMU Munich
Munich, Germany
info@sven-mayer.com

Chris Harrison
Carnegie Mellon University
Pittsburgh, PA, USA
chris.harrison@cs.cmu.edu

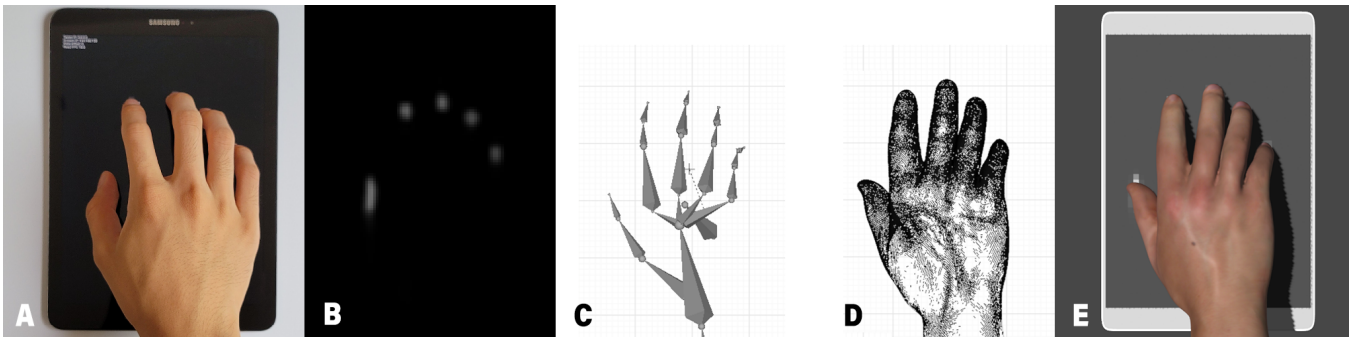

Figure 1: A) Photo reference of hand input. B) Capacitive image as captured by the touchscreen. C) Our software's estimated bone structure for the given input. D) Fitted hand mesh with applied bone weights. E) Final estimated pose and rendered 3D hand.

ABSTRACT

Contemporary mobile devices with touchscreens capture the X/Y position of finger tips on the screen and pass these coordinates to applications as though the input were points in space. Of course, human hands are much more sophisticated, able to form rich 3D poses capable of far more complex interactions than poking at a screen. In this paper, we describe how conventional capacitive touchscreens can be used to estimate 3D hand pose, enabling richer interaction opportunities. Importantly, our software-only approach requires no special or new sensors, either internal or external. As a proof of concept, we use an off-the-shelf Samsung Tablet flashed with a custom kernel. After describing our software pipeline, we report findings from our user study, we conclude with several example applications we built to illustrate the potential of our approach.

CCS CONCEPTS

• **Human-centered computing** → **Gestural input**; **Touch screens**;
• **Computing methodologies** → Reconstruction.

MobileHCI '21, September 27-October 1, 2021, Toulouse & Virtual, France

ACM ISBN 978-1-4503-8328-8/21/09.
https://doi.org/10.1145/3447526.3472045

KEYWORDS

Hand pose, pose estimation, capacitive image

ACM Reference Format:
Frederick Choi, Sven Mayer, and Chris Harrison. 2021. 3D Hand Pose Estimation on Conventional Capacitive Touchscreens. In *Proceedings of the 23rd International Conference on Mobile Human-Computer Interaction (MobileHCI '21), September 27-October 1, 2021, Toulouse & Virtual, France.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3447526.3472045

1 INTRODUCTION

Today's touchscreen mobile computers are marvels of modern engineering, yet touch input remains fairly simplistic, generally only able to report the X/Y position of one or more finger contacts. While this has usefully lowered the barrier to entry for novice users – making interactions "natural" and "intuitive" – it has simultaneously contracted the ceiling of interactive possibilities. For this reason, many complex applications are cumbersome on touch-only devices.

In response, researchers have long looked at ways to move beyond multitouch, and towards more sophisticated and holistic treatment of the hands for user input when operating on a flat-screen. Indeed, as far back as 1976, researchers were exploring touchscreens that could capture not only finger X/Y position, but also X/Y shear force, downwards pressure, and twisting torque for the purposes of a more "natural … rich channel .. for man-machine interaction" [28]. Since then, a host of other sensing techniques and finger input dimensions have been explored, including the angle of attack [44, 69], touch-type [26, 33, 56], digit differentiation [7, 40], and

hand pose estimation [27]. These projects look beyond multi-touch and towards a "rich-touch" future [24].

In this research, we set out to see if conventional capacitive touchscreens (Figure 1A) could be used to estimate a full 3D hand pose (Figure 1C), without any new or external sensors (which is where most research is focused, especially hand-sensing wearables). A live 3D hand model would not only provide the location of finger touches, but also aforementioned input dimensions such as the angle of attack, touch type, and digit differentiation – all in one unified method. In some respects, it is the "holy grail" of touchscreen input, offering a true "digital twin" [11] of the user's hand (Figure 1E). It is this large research vision to which we contribute a new method and proof-of-concept implementation.

While the utility for a live 3D hand model in traditional touchscreen interfaces (2D menus, buttons, sliders, etc.) is currently limited (i.e., existing touch pipelines are sufficient for traditional widgets), we see the growing importance of 3D modalities as an impetus for renewed attention in this field. Mobile touchscreen devices increasingly become gateways for three-dimensional content – whether it be games, CAD, GIS, passthrough AR and many other types of software incorporating 3D elements and manipulation. For example, rather than users simply clicking on 3D objects in smartphone-mediated passthrough AR experiences (analogous to tapping on aquarium glass), we envision virtual hands projecting out into 3D scenes, able to manipulate and interact with objects. Similarly, instead of merely rotating and translating virtual on-screen 2D tools (as seen in e.g., TouchTools [27]), they can be 3D and more richly grasped and manipulated. To illustrate some of our ideas, we created a series of small demos, which we describe at the end of the paper. We now move to key related work, followed by a discussion of our implementation and evaluation.

2 RELATED WORK

We now briefly review three highly-related bodies of work. First are systems that sought to improve touchscreen expressivity by enhancing finger input, but do not consider the hands holistically. Next, we discuss the extensive hand pose literature that exists outside the touchscreen realm, most notably external computer vision techniques and various worn devices. Then, more related to our current work, are systems that recognize hand contour on touchscreens, most often discrete "gestures" and using optical, rather than capacitive touch surfaces. We conclude with a more focused discussion on TouchTools [27], which also seeks to enable more sophisticated whole-hand input on touchscreens.

2.1 Rich Finger Input on Touchscreens

Since the advent of touchscreens, researchers have sought ways to capture additional dimensions of finger input [28]. A major breakthrough in recent decades was practical multi-finger-point sensing (i.e., multitouch) [19, 52], but many other input dimensions are possible, including finger contact area [6, 47], part-of-finger detection [26, 33, 56], finger identification [9, 21, 22, 40, 42, 70], finger pressure [5, 10, 18, 32, 51], finger shear force [25], finger roll [54], and finger orientation [44, 53, 59, 62, 69]. Researchers have also shown that touchscreen capacitive images can be used to improve finger centroid accuracy [37]. In contrast to these systems, our goal is to capture a holistic hand model, which encapsulates the relative geometry of the fingers, palm and wrist.

2.2 Non-Touchscreen Hand Pose

Erol et al. [14] presented a comprehensive literature review covering RGB camera-based approaches to estimate hand pose using computer vision. Other camera varieties have also been considered for this task, most notably depth cameras [1, 35, 58]. There are also commercial motion capture systems, such as Vicon and OptiTrack, can be used to track the hands with millimeter accuracy. However, this requires attaching markers to the hand and fingers [16, 41]. To be more consumer friendly, systems such as LeapMotion, Microsoft HoloLens, and Oculus Quest 2 use a vision-based approach that does not require any user instrumentation. Another active area of research are arm- and hand-worn sensing approaches that leverage techniques such as electromyography, electrical impedance tomography [71], air-pressure sensors [34], time-of-flight cameras [63], infrared camera [36] and accelerometers and gyroscopes [49].

2.3 Hand Shape/Contour on Touchscreens

In order to avoid external and accessory sensors, researchers have long looked at ways to better leverage touchscreen data to capture a users hands. In the domain of "tabletop" systems, we often see touch tracking realized using integrated cameras (such as the original Microsoft Surface). Vision-based touch systems generally have a better spatial resolution than capacitive touch sensors and sometimes have limited depth-sensing capabilities as well (due to diffuse optics at the screen-hand interface). There are also touch sensing systems that utilize external cameras and sensors (e.g., [46]) to capture 3D hand pose, but this hardware arrangement is very different than our approach, where all sensing is contained in a consumer mobile device.

The depth needed to accommodate optics (even "thin" systems using complex waveguides) generally means camera-based tabletop computing interfaces are large and heavy, and as a consequence have fallen out of favor with manufacturers, who have almost exclusively turned to capacitive touchscreens. Nonetheless, camera-based systems allowed researchers to capture and utilize hand shape and contour with high precision for more than a decade, with applications ranging from user identification [55], left vs. right hand differentiation [72], and contact-area-based input techniques [8]. Numerous deep investigations of hand "shape" (i.e., 2D contour) gestural input on tabletop systems have been undertaken, each putting forward unique gesture sets and use cases, which serve as a motivational foundation for our current work [8, 13, 17, 43, 60, 64–67]. Indeed, we draw all of our evaluation hand poses from this prior work, as referenced in Figure 3. We note that while 2D contour is expressive, a true 3D model of the hand is fundamentally a high-order representation. For instance, a 3D hand pose can be used to synthesize a 2D contour or 2D touch points. Another important commonality in the aforementioned systems is that interactions are built around the shape of the 2D contour, and do not infer other hand geometry. For this reason, parts of the hand not directly captured in the contour are not involved in interactions (as opposed to our demos, where "unseen" parts of the hand geometry not touching

the screen, such as the palm or back of the hand, can interact with digital elements, such as in 3D physics simulations).

Most related to our work are systems that capture the geometry of the hand using a conventional capacitive touchscreen. For instance, CapAuth [20] and Mayer et al. [45] had users place their hands flat on a screen to capture a handprint for user authentication and differentiation purposes. While these systems do capture the whole hand geometry in this specific pose, no other poses are considered and no 3D model of the hand inferred for interactive input. PalmTouch [38] also uses a standard touchscreen, using the capacitive image for enable palm touches as a gestural input mode that functions alongside standard finger input. Finally, BodyPrint [31] also uses the capacitive image of a smartphone to detect hand grips, but does not produce a hand model.

Perhaps most similar to our work is TouchTools [27], which argued that a user's familiarity with real-world tools could be brought into a digital context by having users replicate corresponding real-world tool grasps. It was among the first systems to show how a more holistic view of hand input could enable more sophisticated touchscreen interactions. While our system shares similar motivations, we do not limit ourselves to only grasps, and more importantly, our system computes a holistic 3D hand input representation (which innately models grasp) that could be used to power many applications (in fact, we created a TouchTools inspired demo applicaton). In contrast, TouchTools uses a series of geometric and statistical features to classify among 7 "tools", each with an associated discrete hand pose. There is no higher notion of a hand model, or any continuous pose modeling. For some tools, the constellation of hand points controls a tool's rotation, but in most cases the tool is locked to the centroid of the active finger points. Inputting a gesture outside of the 7 canonical poses will lead to a classification error. Implementation wise, TouchTools does not use a capacitive image, but rather touch points (X/Y coordinate and size) provided by an iPad's touch controller. For this reason, "tools" are built around finger tip contacts (knuckles in the case of one tool), as the iPad screen does not accurately report palm or other large inputs.

3 IMPLEMENTATION

We now breakdown the various steps of our software pipeline, starting with retrieving low-level sensor data and ending with a posed 3D hand.

3.1 Open Source

To permit detailed study of our implementation, as well as facilitate replication and others wishing to explore and extend our approach, we have open sourced our system's code at https://github.com/FIGLAB/3DHandPose.

3.2 Tablet & Capacitive Image

For our proof-of-concept implementation, we used a Samsung Galaxy Tab S2 tablet (Figure 1A). This device features a 9.1" screen, which is perhaps the smallest screen size practical for whole-hand interactions (Apple's 12.9" iPad Pro offers more than twice the screen surface area). This 9.1" screen captures a 37×49 capacitive image (4.0mm pitch; Figure 1B) at roughly 60 FPS in normal operation. This capacitive pixel size is roughly average among touch devices (LG 'G' smartwatch (3.5mm pitch) [69], Samsung S4 smartphone (3.9mm pitch) [69], Nexus 5 smartphone (4.1mm pitch) [38], and Microsoft 55" Surface touchscreen (5.9mm pitch) [68]), and thus serves as a reasonable exemplar. Consumer touchscreen devices rarely report their underlying capacitive image data (most often used for debugging), and so we recompiled the open source Android kernel with a custom touchscreen driver (cf. [31, 38, 39]) that is able to communicate directly with the tablet's Synaptics touch controller over i2c (400kHz). Our driver code can be found here *anonymized_for_review*. This i2c interface allows us to stream capacitive images from the touchscreen to the main application processor at ~16 FPS (in parallel with the 60 FPS touch point data). We were able to access this functionality without support from Synaptics; other manufacturers may disable this output, but fundamentally the data exists inside all projected capacitive touchscreen sensors and could be exposed for end-user applications.

While our prototype used a debugging interface, a commercial-grade implementation with support from touchscreen controller and device manufacturers would undoubtedly utilize a high-speed bus able to stream the capacitive image at its native 60 FPS. The data overhead is very small compared to e.g., smartphone cameras or microphones (just 37×49 8-bit values). Although we could have created custom high-speed touchscreen hardware, we felt it was more impactful to demonstrate feasibility using off-the-shelf consumer hardware.

3.3 Normalization

Upon receiving a capacitive touch image from the touch controller, our first step is to normalize the input. This is useful to mitigate variation due to grounding effects (cf. [19]). First, we suppress noise by zeroing all pixels less than 25 in value (the touchscreen

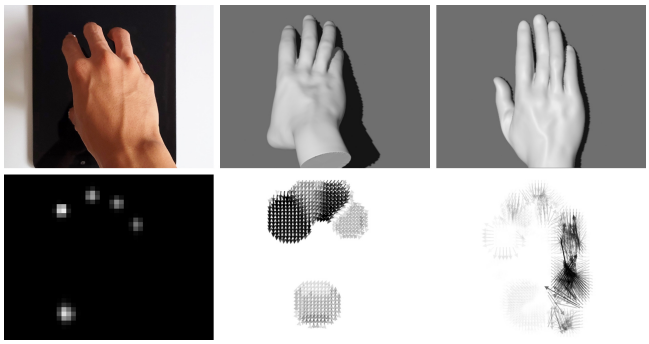

Figure 2: Left top: Photo reference of hand input. Left bottom: the capacitive image captured by the touchscreen. Center column: the best-matching reference pose (top) and its corresponding displacement field (bottom); note all arrows are short, visualizing that very little displacement is needed. Right column: a poorly-matching reference pose (top) and its corresponding displacement field (bottom); note the many long arrows that show large displacements are needed.

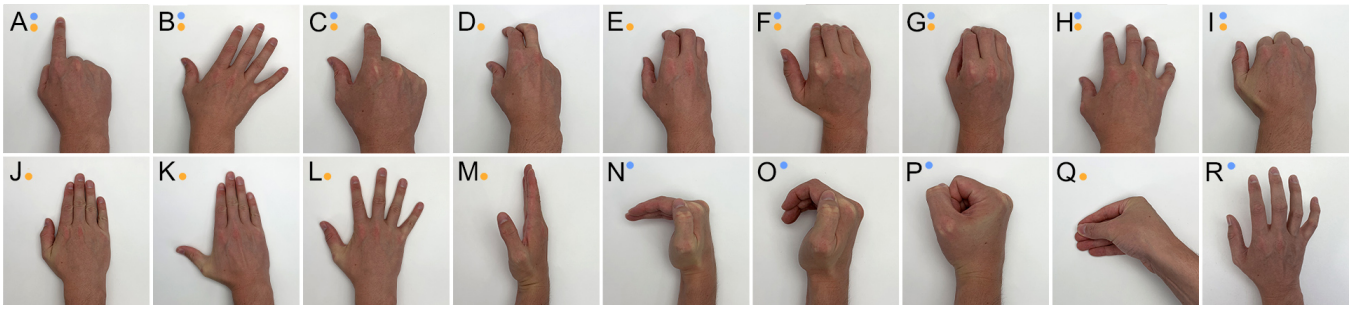

Figure 3: Our hand pose reference set is drawn from the literature: (A) Index finger [8, 13, 17, 40, 46, 60, 65, 66], (B) Thumb [40, 66], (C) Standard pinch [8, 13, 17, 27, 30, 66, 67], (D) Pinch with three fingers [13, 27, 65], (E) Grasping whiteboard eraser [27], (F) Four fingers in a row [8], (G) All fingers together [13, 60, 65], (H) All fingers apart [8, 17, 30, 46, 60, 65, 67], (I) Flat fist with fingers tucked in [13], (J) Hand flat with fingers together [13, 17, 43, 60, 65, 67], (K) 'Hand flat, fingers together with thumb extended [8, 13, 17, 67], (L) Hand flat with fingers spread [13, 17, 30, 66], (M) Chop gesture [8, 13, 17, 60, 66], (N) Corner gesture [8, 13, 17, 43, 66], (O) Hook gesture [13], (P) Fist [13, 17, 60, 67], (Q) Grasping magnifying glass (back of fingers) [27], and (R) Resting palm [38]. A blue dot denotes the pose can be performed without the wrist/palm contacting the screen, while an orange dot denotes the pose can be performed with the wrist/palm contacting the screen. Some poses can only be performed one way.

reports unitless 8-bit unsigned integers). Human touch contacts are significantly higher in capacitance than sensor noise, and so this has the effect of leaving touch points Figure 2 and contact blobs Figure 8 on a relatively clean background (see also Video Figure) for later processing. We then normalize the remaining pixel values linearly between 0 and 255. To standardize translation, we fit an axis aligned bounding box around active pixels and translate the cropped region to the center of a new 50x50 normalized image. For rotation correction, we use the axis along which active pixels exhibit the most variance, estimated using the principal eigenvector of the covariance matrix. Finally, in order to handle different sized hands, a user-specified scale factor can be also applied.

3.4 Reference Hand Pose Library

We initially attempted to programmatically generate a library of possible hand poses by performing a parameter grid search using a rigged hand and IK solver. However, the 27 degrees of freedom of the hand [12] meant the output space was enormous, and even if we could generate all possible poses, it would be computationally challenging to search in real time. Additionally, in practice, we found that many of the generated output poses were artificial, despite human hands being theoretically capable of forming a pose (some examples in Figure 4). A second issue with our purely synthetic approach stemmed from shader code we wrote that converted a generated 3D hand pose into a synthetic capacitive image (for matching, described next). We found that the vast majority of randomly generated hand poses have digits that are bent or lifted from the screen, which meant hundreds of poses would have near-identical capacitive images (example offered in Figure 5) with no clear way to select a "winning" pose.

This outcome significantly influenced our decision to craft a more informed set of reference hand poses, ones that more closely match how humans actually shape their hands to engage in a touchscreen-mediated experience. For this, we turned to the literature, surveying 13 papers from which we drew 18 hand "gestures" and poses. Many of these can be performed two ways – with the palm hovering or resting on the screen – and so several of the poses shown in Figure 3 are functionally considered two poses in our reference set (for a total of 26). To build this library, we had four users (separate from our later study participants; hands ranged between 17 and 19.5 cm in size) perform these hand poses, and we recorded the capacitive image from the touchscreen. The last step is to link the capacitive image with a corresponding 3D hand pose, which we manually pose in Blender [3] using LibHand [61], an open source hand model.

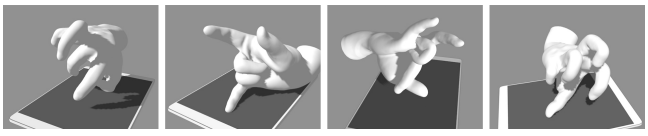

Figure 4: Four example hand poses that demonstrate the limitations of a grid search approach. While all poses are theoretically possible to form, they are rare or near-impossible to hold.

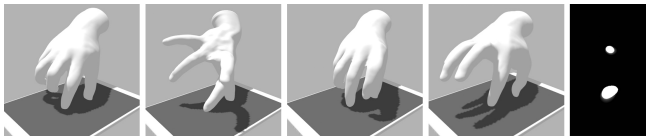

Figure 5: Illustration of different hand poses (left four images) producing equivalent capacitve signals (example shown far right).

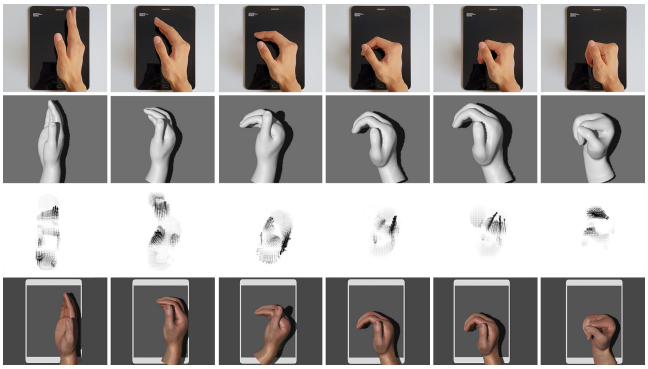

Figure 6: Six snapshots showing a hand transitioning from an chop (pose M) to a fist (pose P). First row: reference photos. Second row: closest reference pose. Third row: displacement field between the live input and best pose. Bottom row: final 3D hand output pose.

3.5 Non-Linear Deformation & Matching

The next step is to take live hand pose input and match it to one of our reference hand poses. We found that the capacitive image of a real hand is rarely an exact match to our pre-recorded reference pose set's capacitive images. For this reason, a simple image difference metric will generally fail. Instead, to better account for geometric variation, we compute the warp between the input capacitive image and all reference hand pose capacitive images. For this, we use Farneback optical flow [15], which is computationally lightweight. This produces a displacement field, with warp vectors for every pixel in the capacitive image. This field can also be thought of like a two-dimensional edit distance - translating, deleting or copying pixels in order to match one capacitive image to another. As such, we found summing the magnitude of the field vectors to be an excellent measure of distance, and we use it as the primary factor in our matching metric. Figure 2 offers an example input alongside a good and poor match, with displacement fields shown for illustration. We found this process to be robust to variation in contact pressure (which tends to enlarge the contact area, but not deform the shape sufficiently to cause an incorrect pose match) and user hand size (once normalized).

An additional issue is that large hand features (notably the palm or side of hand) will activate more pixels, and thus induce more displacement vectors that contribute to the total cost. This can greatly increase the distance to the correct pose match even if all the fingertips match well. To mitigate this effect, we identify all blobs of active pixels (with 4-connectivity). Before we sum the magnitudes of the displacement vectors, we scale the magnitude by the inverse of the number of active pixels in the blob (or closet blob if the vector originated outside active pixels). Additionally, since the displacement field can also delete or create blobs with a comparably low cost, we add an error term that accounts for any difference in blob count. The final cost for matching is then:

$$10 \times \Delta\text{blob count} + \sum_{v \in \text{flow vectors}} \frac{\|v\|}{W_v}$$

where W_v the number of active pixels in the blob closest to the origin of the flow vector v.

3.6 End Effector Manipulation and Inverse Kinematics

The 26 discrete hand poses in our reference set is much too coarse to smoothly animate or depict fine-grained hand movements. In other words, if we used only these poses, any dynamic hand input would appear to instantly snap between reference poses, breaking realism and precluding many interesting inputs. Thus, once a match is made to a coarse reference pose, we perform a second-stage fine manipulation of the 3D model to better match the live hand pose.

The open source hand model [61] we use contains 21 joints in a fully articulating skeleton (Figure 1). We specify six end effectors – the five fingertips and the wrist. We use Blender's inverse kinematics (IK) solver [4] to resolve natural hand poses given our input data. More specifically, we project the winning reference pose's rigged hand down onto our aforementioned displacement field, which maps the winning reference pose to the live hand input. The displacement vector that maps to each end-effect's position is then applied as a fine-grained warping. One can think of this process similar to an elastic sheet, which has been non-linearly stretched, and then pinning a reference rigged hand to the sheet and releasing the tension - the bone structure of the rigged hand will now "snap" to some intermediate state.

This result of this process not only yields hand poses that better match user input, but also allows for continuous animation between our sparse reference hand pose set. Figure 6 offers an illustrative example showing a user moving between 'chop' (pose M) and 'fist' (pose P) — transitioning through poses N and O – along with the corresponding displacement vector field and final hand pose output.

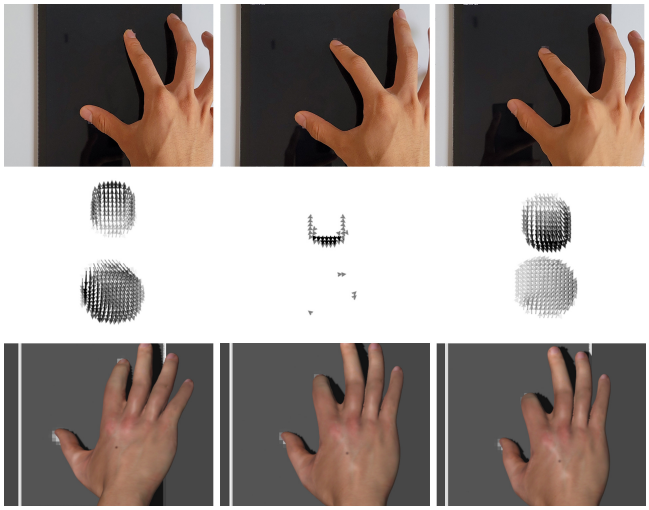

Figure 7: Three different inputs of a pinch gesture (pose C). First row: reference photos. Second row: displacement field between the live and best reference pose. Third row: rendered 3D hand pose. Note how the distance between the pinching fingers is faithfully captured.

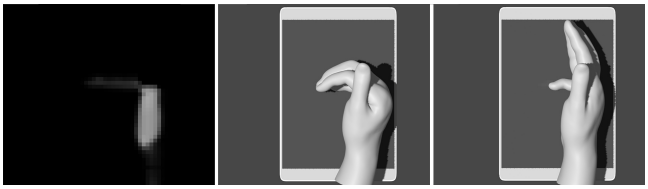

Figure 8: Example of additional pose logic. Although only the pinky is captured in the touchscreen capacitive image (left), we decide to animate all of the fingers as one unit (center), as opposed to just moving the pinky alone (right), as this is less natural.

Figure 7 offers a second example, where a two-finger pinch (pose C) is being warped to more faithfully match the user input. The other significant benefit of this pose interpolation is our ability to dramatically reduce the size of our reference library to key poses, allowing for greater computational efficiency. This approach stands in contrast to e.g., [58], which used a very large library of intermediate hand poses (30K) and extreme parallelization to identify a candidate hand pose for its computer-vision-driven approach.

3.7 Special Pose Heuristics

For hand poses where the palm and fingertips rest on the screen, there is sufficient data for the IK solver to produce a realistically posed output. However, other hand poses have parts lifted from the screen; while the IK solver will try its best given the constraints, some outputs are patently incorrect, while others are unstable. To improve pose realism and stability when only limited data about the hand is available, each of our reference hand poses can include optional posing heuristics that introduces additional positional constraints. For instance, with our 'L' hand pose, only the pinky finger is captured by the touchscreen. While we could animate the pinky finger independently, we observed it is far more likely for users to move their fingers together as a unit (Figure 8, center), rather than just the pinky alone (Figure 8, right), and so this reference pose contains this additional logic. Similarly, for all hand poses with one or more digits raised, we do not know the true height of the hovering fingers, and so we simply set their target position to between 2-6 cm above the surface of the screen (depending on the pose) as a coarse estimate for the IK solver.

3.8 3D Hand Output & End-User Applications

We use Blender's armature deform parenting to realistically skin a hand mesh in accordance with the underlying skeleton, now fully posed by our pipeline. LibHand [61] also comes with a photo-realistic skin texture, which can be applied if desired. We envision our 3D hand pose estimates being exposed via an API on touchscreen devices. In its simplest form, applications developers could instantiate event listeners for hand poses in much the same way as basic touch events today; e.g., instead of onTouchDown(), it might be onFistBump(). For uses in games or mobile augmented reality, the full 3D hand mesh could be requested and rendered on-screen. As a proof of concept, we used the current version of our API to build several example applications, described later.

3.9 Performance

Our implementation described above was designed to be a vehicle for rapid exploration, experimentation and iteration. It is written in Python and JavaScript – interpreted languages not known for their performance. We also rely on a suite of software packages, such as Blender, and pass data around using local sockets. In short, this is how one designs a research proof-of-concept, but not a commercial product. Nonetheless, we can report the performance on two different machines that we used for development: On a 3.8GHz Ryzen 9 3900X desktop, our process takes up less than 5% of the CPU and runs as fast as the tablet can transmit capacitive images over USB (16.2 FPS). On a less powerful Intel Core i7 laptop, the full stack runs at ~10 FPS consuming around 25% of the CPU.

More interesting is to consider how our pipeline might run on mobile hardware if it were to go through a commercial engineering process. Rather than extrapolate, we can look to a close analog: the 3D hand pose tracking available in the Oculus Quest 1 and 2. Facebook's software has to deal with more pixels than our capacitive image approach, but nonetheless must also pose a 3D hand model containing 21 joints (same as our system). As noted in the software's academic paper [23], the hand pose estimation runs at 30 FPS on a Qualcomm Snapdragon 835 mobile processor with 4GB of RAM (roughly equivalent to an iPhone 7 in performance), and of course there is extra compute available to run intensive VR games for hours on battery power. The newer Oculus Quest 2 is even faster, and today's flagship phones are multiples faster. This demonstrates what commercial engineering and optimization can achieve, and there is no fundamental reason that similar performance is not possible with our approach.

4 EVALUATION

We gathered quantitative and qualitative data from 12 participants to evaluate our system's ability to estimate 3D hand pose. We now describe our apparatus and procedure, followed by main results.

4.1 Apparatus

Participants were asked to sit in front of our tablet resting flat on a table. For data recording, the tablet was connected to a desktop PC via USB (grounding the tablet and also keeping it charged), streaming capacitive image directly to our software pipeline. Note that our software can also operate untethered and ungrounded, as seen in our Video Figure (see e.g., capacitive images when plugged in (0:23-0:36) and not (0:36-0:45), and later in the video, demos running with/without grounding). A separate monitor was used to prompt poses from participants, and was also used to show the estimated pose result, allowing them to rate the output quality. Although we could texture the hand, we wished to minimize the effect of color and gender [57], and so hands were presented in a flat gray. An private experimenter's view running on another monitor allowed us to guide the participants through the stages of the study. Importantly, the experimenter did not see the live pose output until a trial was captured (to avoid bias).

As the front and back of users' hands were used to touch the screen across our various poses, it was not possible to attach infrared reflectors to the hand for use with a high-precision motion tracking system. As an necessary experimental compromise, we

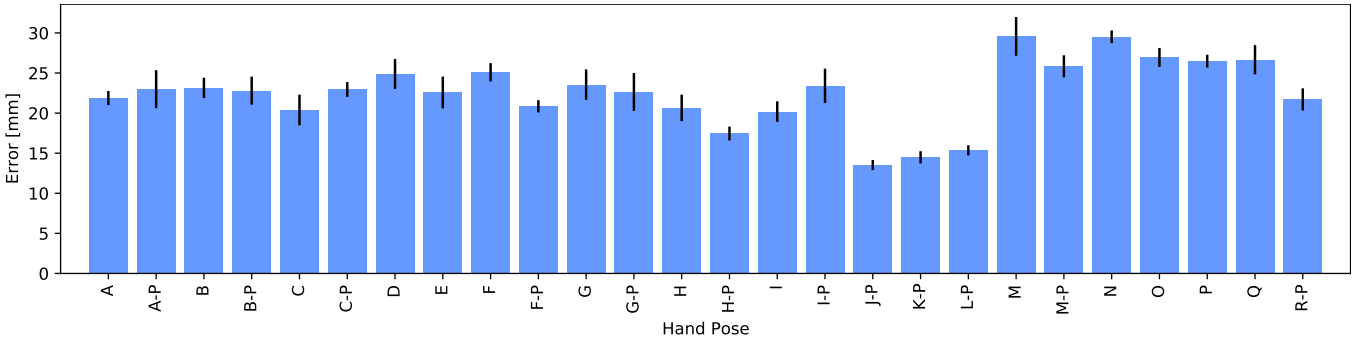

Figure 9: The joint displacement for the 26 references poses. See Figure 3 for pose letter key. '-P' means the pose was with the palm contacting the screen surface.

used a Microsoft Kinect mounted 1m above the tablet to capture depth maps and calibrated RGB stills of participants' hand poses. We experimented with automated hand keypoint labeling [48], but found the output to be occasionally catastrophically incorrect (requiring manual review) and even when "correct", spatial error of joints was substantial (even though the hand skeleton was correct holistically speaking). We also tried a Leap Motion, but it was confused by the tablet reflecting infrared light (the device is designed to illuminate the hands in the air without strong environmental reflections). Instead, joints were manually annotated after the study by personnel without knowledge of the experiment, providing a 3D ground truth for benchmarking.

4.2 Procedure

After welcoming participants, participants were given a brief orientation. We answered all open questions and asked them to sign an consent form. Afterwards, we asked participants to place their hand onto the screen in order to capture the length of their hand (from wrist to farthest finger tip), which was used to automatically scale all pose outputs. Participants were then prompted to perform all 26 poses in our reference pose set (Figure 3) in a random order. This was done by showing them a reference photograph (Figure 3) on an external monitor. Once the participant was satisfied they had assumed the pose, the participant informed the experimenter, and a single snapshot of the 3D hand pose from our live pipeline (which was running continuously) was saved in concert with the aforementioned Kinect depth and RGB data. This hand pose data was then immediately shown to participants in a top-down view (example output shown in Figure 6, bottom row). While maintaining their hand pose, participants were then asked to verbally answer two questions shown on-screen using a 7-point Likert scale (strongly disagree to strongly agree). The first statement was "The virtual representation of my hand matches my real hand's pose", which sought to answer whether the virtual hand rendition matched their actual hand pose. The second statement was"The virtual representation of the hand looks natural", which we included to measure if the virtual hand was posed in a natural manner, even if it did not match the user's hand pose (i.e., that our IK was able to solve for a reasonable pose, even in cases of a tracking or posing failure). Both questions were reviewed with participants during the study orientation to confirm their understanding. Once both questions were answered, users could withdraw their hand and relax. When they indicated they were ready to proceed, a new trial began by once again showing a reference photograph of a requested pose. The study took around 20 minutes to complete, and participants were compensated USD10 for their time.

4.3 Participants

We recruited 12 participants (2 female) from our institution, with a mean age of 25.8 (SD=4.7). As noted previously, none of the 12 participated were used in capturing reference data for our hand pose library. The mean length of participant hands was 18.0 cm (SD=1.1) and the mean width was 8.5 cm (SD=0.8). These span from the 5th to 95th percentile of human hand sizes [50]. Other than asking participants to form a pose, we explicitly chose not to control for aspects such as pressure to capture variability in our data and begin to explore generalizability across users.

4.4 Results

To evaluate the spatial accuracy of our pipeline, we compare the 3D position of each virtual hand joint to the annotated, real-world user's hand. As we care about the relative arrangement of the hand joints that make up a pose (as opposed to absolute world position), we use a palm-origin coordinate system.

Across all poses and participants, we found a mean 3D euclidean joint error of 22.4mm (SD=5.9mm). Figure 9 provides an error breakdown for the 26 hand poses we tested. For poses where many joints touch the screen (e.g., poses J, K, and L) error is 14.5mm (SD=2.1). If we look only at joints in contact with the screen (i.e., cases where we have direct sensor data), we find a mean euclidian error of 11.2mm (SD=9.8mm) across all 26 poses. This means that end effectors that are likely to be directly interacting with digital content are also the most accurate. In contrast, poses that have just one or two joints touching the screen (e.g., poses A and B) have much higher error due in large part to the system having to estimate totally un-sensed joints.

This effect is illustrated in our Video Figure. For example, at 1:40-1:42, where a user performs a drag with the "index finger" (A) pose, the one joint physically touching the screen (index finger tip) is stable and accurate, while the other 14 joints have to be

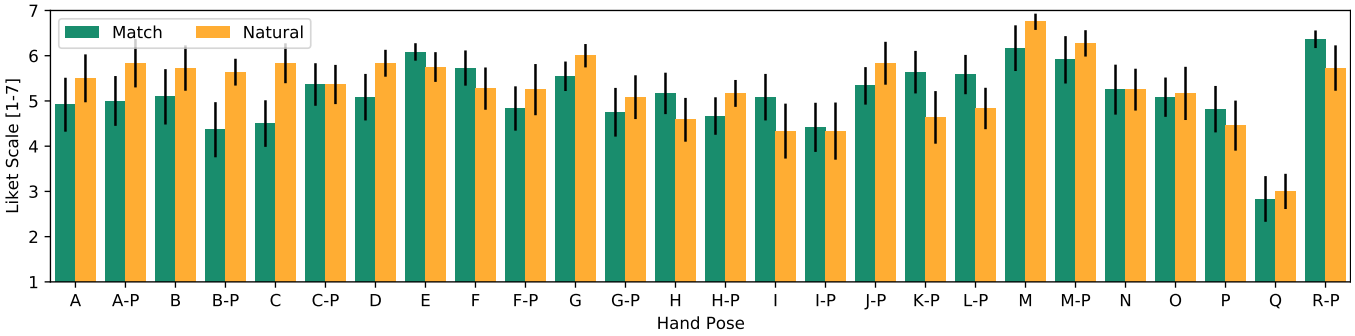

Figure 10: The "match" and "natural" Likert ratings for the 26 references poses. See Figure 3 for pose letter key. '-P' means the pose was with the palm contacting the screen surface.

guessed purely off the estimated pitch and yaw of that one joint, which is inherently accurate, especially as one moves along bone linkages. Next in the video (1:42-1:44), a "resting palm" (R) pose is performed, where once again a vast majority of the hand joints are being estimated in 3D space with no direct data (just inferences). In both pose cases, if our system misestimates the hand angle by even 10 degrees, it will lead to centimeter-level inaccuracies at distant joints (driving up mean error). Likewise, if the user bends their fingers away from our guessed neutral pose, this will again lead to compounding spatial errors. Nonetheless, the overall gestalt of the hand pose is preserved. Our Video Figure provides a good sense of how this error manifests in practice.

In a similar manner, we found that poses that orient the hand perpendicular to the screen (e.g., poses M, N, O and P) had higher error (mean=27.7mm, SD=4.4), stemming from having to estimate Z distance of joints (i.e., distance from the touchscreen's surface) with very little information (e.g., fingers held tightly together or spread apart). Interestingly, we did not find a correlation between the quantitative correctness (joint spatial error) and qualitative correctness (hand match questionnaire), with a Pearsons correlation factor of -.063.

We note that even though our system stores a library of reference hand poses internally, it is not valid to simply compute discrete pose recognition accuracy. This is because our large pose set (compared to any prior work) is meant to enclose a continuous pose space, and as such, one pose can often be warped to another. For example, poses E, G and H can be warped to one another, as can J/K/L and M/N/O/P. In many cases, our optimization function prefers to warp an "incorrect" pose to achieve a better geometric match, and as such, mean euclidean join error is a superior evaluation metric for a system estimating 3D hand pose.

We found that all of our participants had roughly the same performance, with no outliers, suggesting that hand size, pressure, skin moisture, and user grounding condition are not major factors, though we caution that n=12 is small. However, we note this matches our anecdotal observations throughout many months of development, which involved more individuals. The fact is these touchscreens have been highly engineered for robust performance across a wide range of users and environments (e.g., high/low humidity environments and adverse conditions such as rain drops on the screen). Our pipeline was designed to account for variations in contact condition – whether that be from different hand sizes (we tested 5-95th percentile) or touch pressure – by looking for the closest matching pose.

For the question about the virtual hand "matching" their real hand, we found that all gestures but pose Q ("Grasping magnifying glass") had a positive rating above "neutral" (>4). Pose Q is the only pose where the back of the hand touches the screen and we hypothesize that the boniness of this part of the body precludes good contact with the capacitive touchscreen, and results in an unreliable signal for pose estimation. Across the remaining 25 poses, the average quality score was 5.2 out of 7 (SD=0.5); see Figure 10. For the question about if the generated, virtual hand looked "natural", we received similar responses. Again all but pose Q was rated higher than 4 (neutral) by our participants (mean quality score of 5.4, SD=0.6; Figure 10). This suggests that our IK solver was good at preventing unnatural hand poses, even when it received confusing or conflicting data. To understand how the two Likert ratings were effecting each other, we first performed a Wilcoxon signed-rank test. The test showed no significant differences (Z = 8184, p=.230) between the two questions; a Pearsons correlation showed a correlation of .464.

5 EXAMPLE APPLICATIONS

We believe the next step beyond multitouch is a fully-realized, real-time, 3D model of the user's input hand. This could enable a wealth of rich input techniques, from finger-level interactions with dimensions such as angle of attack, all the way to whole-hand manipulations. To help illustrate this generalizability, we selected four exemplary application areas that could benefit from our approach. These applications are also demonstrated in our Video Figure. We include a fifth section with other speculative uses.

5.1 3D Physics & Manipulation

As a 3D manipulation demo, we created a simple waterfall simulation where droplets can flow around the complex geometry of a hand reaching into the scene (Figure 11A). In a similar vein, we created a playground of blocks (Figure 11B) that users can grab and push around, taking into account full 3D hand shape. In both of these example apps, no hand model is seen on the screen. Instead, we translate an invisible copy of the virtual hand model to just

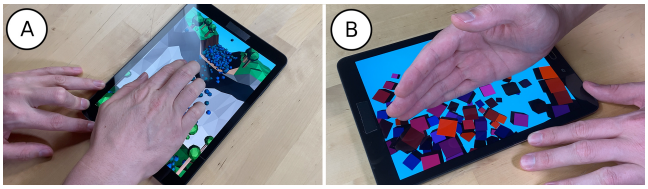

Figure 11: A) a minecraft-esque waterfall simulation has droplets that bounce and flow off the user's actual, full-hand geometry. B) a user interacts with blocks, which conform to the actual 3D geometry of the user's hand pose as if it were translated down into the scene.

below the touchscreen glass, where it can participate in a physical simulation. By using an orthographic projection (and assuming the user is looking roughly perpendicular to the screen), we can create a perspective illusion that the user's real hand is interacting with virtual objects (e.g., virtual droplets appear to bounce off the user's actual knuckles).

5.2 BumpTop++

Agarawala and Balakrishnan [2] presented the idea of a physics-based computer desktop environment, later called BumpTop. In this 3D simulation, users could push files around, make piles, and perform similar physical manipulations. However, input was through a stylus — inherently single-point and two-dimensional - which limited the otherwise rich 3D experience. Inspired by this creative research, we created a BumpTop-inspired clone leveraging our 3D hand pose pipeline. This allows users to not just form piles with their bare hands, but also pickup items with their fingertips and create stacks in a much more natural and inherently 3D manner, see Figure 12.

5.3 TouchTools++

Another inspirational research system was TouchTools [27]. In this system, users could form different hand grasps, place that hand down onto a touchscreen, and a corresponding tool would be summoned (e.g., marker, whiteboard eraser, magnifying glass). The tools, however, were entirely 2D – both graphically and in how they could be manipulated. For example, the marker tool could be translated and rotated on the screen, but not tilted. Of course there are many tools where true 3D manipulation is key to their expressively, such as a chisel-tip marker or paintbrush. To illustrate this, we created a simple TouchTools-inspired demo that not only

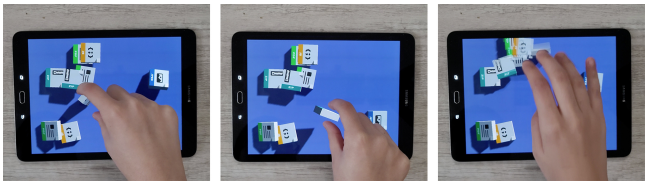

Figure 12: In this BumpTop-inspired demo [2], users can pick up files (left & center), create piles, and push documents around (right).

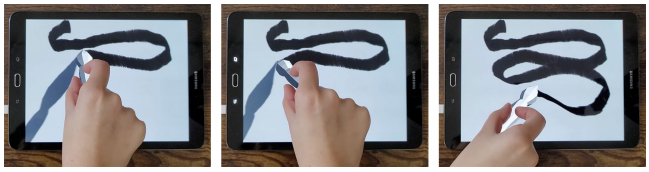

Figure 13: In this TouchTools-inspired demo [27], the hand mesh is used to manipulate virtual tools, such as this brush, with six degrees of freedom instead of the usual two, offering more expressive input.

summons a tool based on the hand grasp, but then also correctly manipulates that tool in 3D. Figure 13 shows an example sequence of someone drawing calligraphy with a brush, where the tilt impacts the stroke thickness.

5.4 "Reach Into" Passthrough Mobile AR

In our final example, We envision mobile AR applications that allow users to "reach" into scenes by projecting the user's 3D hand pose out in front of the device. In contrast to just tapping on the screen glass as if it were a mere window looking out onto a scene full of 3D objects, our approach could be considerably more immersive and embodied. To illustrate this new interaction paradigm, we created a simple educational game (Figure 14) where children must identify virtual recyclable waste distributed in an environment, pick these items up, and deposit them into a recycling bin using appropriate hand grasps.

5.5 Other Speculative Uses

Beyond the five examples applications we built, there are many other uses of 3D hand pose data on touchscreens. We look forward to building these apps and submitting them as part of future publications, along with context-specific evaluations. However, we provide a brief overview of some of the ideas we have planned to illustrate future and speculative uses.

For example, text entry on touchscreen devices continues to be a significant input bottleneck. Without good tactile references, the fingers tend to drift or mis-click keys, leading to errors. However, knowledge of what finger is touching the screen could be used to help resolve ambiguities in typed input. For example, if the touchscreen sees a touch event between the 'g' and 'h' keys, it may not be entirely obvious which letter to trigger. However, by

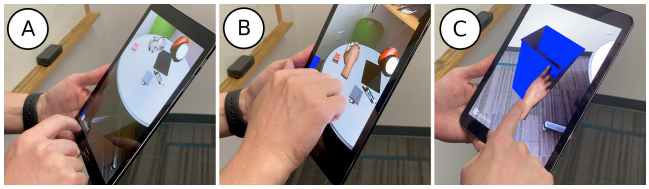

Figure 14: In this educational mobile AR game, users must identify recyclable waste (A), pick it up with a virtual hand projected into the scene (B), and deposit it into a recycling bin (C).

knowing what digit spawned the touch event (e.g., left or right index fingers), a bias could be applied. In this example, it is more likely the left index finger would type a 'g' than a 'h' (even if the Cartesian coordinates of the touch event happened inside the 'h' key's hit box). In fact, each finger could have a custom key-region mapping to improve accuracy.

We also believe there are many creative uses in gaming, which can bring novelty and fun to touch experiences. For example, in a fantasy game, the user's hand input could be used to cast spells or wield different physical weapons. Building sims could allow users to grasp, stack, hammer and saw materials, again using appropriate grasps and hand poses. Like with our pass-through augmented reality recycling app, users could also reach into new gaming experiences, such as physically trying to catch Pokemon.

Finally, we also foresee uses in social apps and experiences. In reality, human-human communication includes expressive hand use. As a simple example, messaging apps could trigger hand gesture emoticons by the user physically performing that hand pose on the screen (e.g, high five, thumbs up). Or perhaps user can send animated hand gesture messages - almost a handimoji. In social experiences employing avatars, a user's hands could be more dynamically expressed than simply pressing a compresspond hand gesture button.

6 LIMITATIONS & FUTURE WORK

While we believe our pipeline demonstrates feasibility, it is not yet sufficiently accurate for consumer applications. This gap is perhaps magnified by the impressive performance of modern touchscreens, which are both accurate and low latency, setting a high bar. Our computational requirements are much higher than standard touch pipelines, where almost all computation (adaptive backgrounding, blob segmentation, touch tracking, etc.) occurs on a dedicated touch controller IC, with only processed touch events being passed to the main application processor. For the foreseeable future, we envision our pipeline having to run on the main application processor, which is more power intensive. For this reason it would seem unlikely for our process to run continuously, as the standard touch pipeline does, and instead would start as a background service when applications request such data (not unlike other computationally expensive APIs, such as opening a camera or running an AR SDK). As discussed in greater length in our Performance section, our pipeline is very much a proof-of-concept implementation, and is not presently able to run on mobile devices. However, as also noted in that section, there are analogous processes that have gone though commercial-level engineering efforts that show that equivalent complexity is possible on mobile-grade hardware, which continues to make impressive strides in performance.

We also note that while our twelve-participant study demonstrates the overall feasibility of our approach, there are obviously a range of factors that merit future investigation. For example, we did investigate children's hands (our participant hand sizes only ranged from 5-95th percentile of adult hands). Likewise, the ability for a user to place their hand on a screen at a tilt, or even while on-the-go, will no doubt influence the capacitive image. Both conditions are readily detected by a mobile device's IMU, so perhaps different reference pose sets (or match parameters) could be loaded to compensate. In adverse conditions, such as raindrops on the screen or a user with wet hands, the capacitive image will vary and require other compensation strategies. Additionally, we only evaluated stable hand poses. While our pose test set is large compared to prior work (e.g., TouchTools' [27] 7 poses vs. our 26 poses), it does not evaluate dynamic tracking accuracy (e.g., hands moving and inter-pose classification) – an experimental compromise given the limits of uninstrumented external 3D hand tracking technologies (see Procedure section). Instead, we modeled our study procedure on that found in prior HCI hand pose work such as [17, 27, 43, 46]. That said, our Video Figure demonstrates continuous and inter-pose hand tracking, which offers a point of reference.

Finally, despite capacitive touchscreens only providing a very coarse sensor image (with capacitive pixels around 4mm in size), this resolution is sufficient to capture the geometry of the hand, including the smallest element we have to handle: fingertips. This is not coincidental – touchscreens have been very carefully engineered to be the lowest possible resolution (chiefly to improve scan rate) while still able to accurately capture finger input. For this reason, we suspect that higher-resolution touchscreens would provide only marginally better hand poses. That said, there is one dimension of touchscreen data where improvement would be welcome: sensing range. If this could be extended to several centimeters, as demonstrated in Hinckley et al. [29], it would provide much more 3D data about the shape of the hands not immediately in contact with the surface of the screen.

7 CONCLUSION

We have described our system that allows conventional 2D capacitive touchscreens to infer a user's 3D hand pose. We believe this capability will be important as mobile touchscreen devices increasingly become gateways for three-dimensional content – whether it be games, CAD, GIS, passthrough AR and many other types of software incorporating 3D elements and manipulation. Uniquely, our approach is software-only and could be made to run on any modern capacitive touchscreen device. While significant work remains, we believe our software pipeline demonstrates imminent feasibility, and we hope others will join us in fully exploring and enabling this input modality in future work.

REFERENCES

[1] Farid Abedan Kondori, Shahrouz Yousefi, Jean-Paul Kouma, Li Liu, and Haibo Li. 2015. Direct hand pose estimation for immersive gestural interaction. *Pattern Recognition Letters* 66 (2015), 91 – 99. https://doi.org/10.1016/j.patrec.2015.03.013

[2] Anand Agarawala and Ravin Balakrishnan. 2006. Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles and the Pen. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada) *(CHI '06)*. Association for Computing Machinery, New York, NY, USA, 1283–1292. https://doi.org/10.1145/1124772.1124965

[3] Blender Online Community. 2020. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam. http://www.blender.org

[4] Blender Online Community. 2020. *Blender - Introduction to Inverse Kinematics*. Blender Foundation, Blender Institute, Amsterdam. https://docs.blender.org/manual/en/latest/animation/armatures/posing/bone_constraints/inverse_kinematics/introduction.html

[5] Tobias Boceck, Sascha Sprott, Huy Viet Le, and Sven Mayer. 2019. Force Touch Detection on Capacitive Sensors Using Deep Neural Networks. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services* (Taipei, Taiwan) *(MobileHCI '19)*. Association for Computing Machinery, New York, NY, USA, Article 42, 6 pages. https://doi.org/10.1145/3338286.3344389

[6] Sebastian Boring, David Ledo, Xiang 'Anthony' Chen, Nicolai Marquardt, Anthony Tang, and Saul Greenberg. 2012. The Fat Thumb: Using the Thumb's Contact Size for Single-handed Mobile Interaction. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services* (San Francisco, California, USA) *(MobileHCI '12)*. ACM, New York, NY, USA, 39–48. https://doi.org/10.1145/2371574.2371582

[7] Drini Cami, Fabrice Matulic, Richard G. Calland, Brian Vogel, and Daniel Vogel. 2018. Unimanual Pen+Touch Input Using Variations of Precision Grip Postures. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (Berlin, Germany) *(UIST '18)*. Association for Computing Machinery, New York, NY, USA, 825–837. https://doi.org/10.1145/3242587.3242652

[8] Xiang Cao, A. D. Wilson, R. Balakrishnan, K. Hinckley, and S. E. Hudson. 2008. ShapeTouch: Leveraging contact shape on interactive surfaces. In *3rd IEEE International Workshop on Horizontal Interactive Human Computer Systems (TABLETOP 2008)*. IEEE, 129–136. https://doi.org/10.1109/TABLETOP.2008.4660195

[9] Ashley Colley and Jonna Häkkilä. 2014. Exploring Finger Specific Touch Screen Interaction for Mobile Phone User Interfaces. In *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: The Future of Design* (Sydney, New South Wales, Australia) *(OzCHI '14)*. ACM, New York, NY, USA, 539–548. https://doi.org/10.1145/2686612.2686699

[10] Christian Corsten, Simon Voelker, Andreas Link, and Jan Borchers. 2018. Use the Force Picker, Luke: Space-Efficient Value Input on Force-Sensitive Mobile Touchscreens. 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, 1–12. https://doi.org/10.1145/3173574.3174235

[11] Abdulmotaleb El Saddik. 2018. Digital twins: The convergence of multimedia technologies. *IEEE multimedia* 25, 2 (2018), 87–92. https://doi.org/10.1109/MMUL.2018.023121167

[12] George ElKoura and Karan Singh. 2003. Handrix: Animating the Human Hand. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, California) *(SCA '03)*. Eurographics Association, Goslar, DEU, 110–119.

[13] Julien Epps, Serge Lichman, and Mike Wu. 2006. A Study of Hand Shape Use in Tabletop Gesture Interaction. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems* (Montréal, Québec, Canada) *(CHI EA '06)*. Association for Computing Machinery, New York, NY, USA, 748–753. https://doi.org/10.1145/1125451.1125601

[14] Ali Erol, George Bebis, Mircea Nicolescu, Richard D. Boyle, and Xander Twombly. 2007. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding* 108, 1 (2007), 52 – 73. https://doi.org/10.1016/j.cviu.2006.10.012

[15] Gunnar Farnebäck. 2003. Two-Frame Motion Estimation Based on Polynomial Expansion. In *Image Analysis*, Josef Bigun and Tomas Gustavsson (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 363–370. https://doi.org/10.1007/3-540-45103-X_50

[16] Anna Maria Feit, Daryl Weir, and Antti Oulasvirta. 2016. How We Type: Movement Strategies and Performance in Everyday Typing. 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, 4262–4273. https://doi.org/10.1145/2858036.2858233

[17] Dustin Freeman, Hrvoje Benko, Meredith Ringel Morris, and Daniel Wigdor. 2009. ShadowGuides: Visualizations for in-Situ Learning of Multi-Touch and Whole-Hand Gestures. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces* (Banff, Alberta, Canada) *(ITS '09)*. Association for Computing Machinery, New York, NY, USA, 165–172. https://doi.org/10.1145/1731903.1731935

[18] Alix Goguey, Sylvain Malacria, and Carl Gutwin. 2018. Improving Discoverability and Expert Performance in Force-Sensitive Text Selection for Touch Devices with Mode Gauges. 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, 1–12. https://doi.org/10.1145/3173574.3174051

[19] Tobias Grosse-Puppendahl, Christian Holz, Gabe Cohn, Raphael Wimmer, Oskar Bechtold, Steve Hodges, Matthew S. Reynolds, and Joshua R. Smith. 2017. Finding Common Ground: A Survey of Capacitive Sensing in Human-Computer Interaction. 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, 3293–3315. https://doi.org/10.1145/3025453.3025808

[20] Anhong Guo, Robert Xiao, and Chris Harrison. 2015. CapAuth: Identifying and Differentiating User Handprints on Commodity Capacitive Touchscreens. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces* (Madeira, Portugal) *(ITS '15)*. ACM, New York, NY, USA, 59–62. https://doi.org/10.1145/2817721.2817722

[21] Aakar Gupta, Muhammed Anwar, and Ravin Balakrishnan. 2016. Porous Interfaces for Small Screen Multitasking Using Finger Identification. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 145–156. https://doi.org/10.1145/2984511.2984557

[22] Aakar Gupta and Ravin Balakrishnan. 2016. DualKey: Miniature Screen Text Entry via Finger Identification. 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, 59–70. https://doi.org/10.1145/2858036.2858052

[23] 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 Transactions on Graphics (TOG)* 39, 4, Article 87 (July 2020), 13 pages. https://doi.org/10.1145/3386569.3392452

[24] Chris Harrison. 2014. The Rich-Touch Revolution is Coming. (2014). https://tei.acm.org/2014/program/keynote_harrison.php at TEI.

[25] Chris Harrison and Scott Hudson. 2012. Using Shear as a Supplemental Two-Dimensional Input Channel for Rich Touchscreen Interaction. 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, 3149–3152. https://doi.org/10.1145/2207676.2208730

[26] Chris Harrison, Julia Schwarz, and Scott E. Hudson. 2011. TapSense: Enhancing Finger Interaction on Touch Surfaces. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. ACM, New York, NY, USA, 627–636. https://doi.org/10.1145/2047196.2047279

[27] Chris Harrison, Robert Xiao, Julia Schwarz, and Scott E. Hudson. 2014. TouchTools: Leveraging Familiarity and Skill with Physical Tools to Augment Touch Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2913–2916. https://doi.org/10.1145/2556288.2557012

[28] Christopher F. Herot and Guy Weinzapfel. 1978. One-Point Touch Input of Vector Information for Computer Displays. *SIGGRAPH Comput. Graph.* 12, 3 (Aug. 1978), 210–216. https://doi.org/10.1145/965139.807392

[29] 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* (Santa Clara, California, USA) *(CHI '16)*. ACM, New York, NY, USA, 2869–2881. https://doi.org/10.1145/2858036.2858095

[30] Uta Hinrichs and Sheelagh Carpendale. 2011. Gestures in the Wild: Studying Multi-Touch Gesture Sequences on Interactive Tabletop Exhibits. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 3023–3032. https://doi.org/10.1145/1978942.1979391

[31] Christian Holz, Senaka Buthpitiya, and Marius Knaust. 2015. Bodyprint: Biometric User Identification on Mobile Devices Using the Capacitive Touchscreen to Scan Body Parts. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. ACM, New York, NY, USA, 3011–3014. https://doi.org/10.1145/2702123.2702518

[32] Kaori Ikematsu, Masaaki Fukumoto, and Itiro Siio. 2019. Ohmic-Sticker: Force-to-Motion Type Input Device That Extends Capacitive Touch Surface. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 1021–1030. https://doi.org/10.1145/3332165.3347903

[33] Kaori Ikematsu and Shota Yamanaka. 2020. ScraTouch: Extending Interaction Technique Using Fingernail on Unmodified Capacitive Touch Surfaces. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 3, Article 81 (Sept. 2020), 19 pages. https://doi.org/10.1145/3411831

[34] Pyeong-Gook Jung, Gukchan Lim, Seonghyok Kim, and Kyoungchul Kong. 2015. A Wearable Gesture Recognition Device for Detecting Muscular Activities Based on Air-Pressure Sensors. *Transactions on Industrial Informatics* 11, 2 (2015), 485–494. https://doi.org/10.1109/TII.2015.2405413

[35] Cem Keskin, Furkan Kıraç, Yunus Emre Kara, and Lale Akarun. 2013. *Real Time Hand Pose Estimation Using Depth Sensors*. Springer London, London, 119–137. https://doi.org/10.1007/978-1-4471-4640-7_7

[36] David Kim, Otmar Hilliges, Shahram Izadi, Alex D. Butler, Jiawen Chen, Iason Oikonomidis, and Patrick Olivier. 2012. Digits: Freehand 3D Interactions Anywhere Using a Wrist-Worn Gloveless Sensor. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge, Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 167–176. https://doi.org/10.1145/2380116.2380139

[37] Abinaya Kumar, Aishwarya Radjesh, Sven Mayer, and Huy Viet Le. 2019. Improving the Input Accuracy of Touchscreens Using Deep Learning. 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/10.1145/3290607.3312928

[38] Huy Viet Le, Thomas Kosch, Patrick Bader, Sven Mayer, and Niels Henze. 2018. PalmTouch: Using the Palm as an Additional Input Modality on Commodity Smartphones. 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/10.1145/3173574.3173934

[39] Huy Viet Le, Sven Mayer, and Niels Henze. 2018. InfiniTouch: Finger-Aware Interaction on Fully Touch Sensitive Smartphones. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (Berlin, Germany) *(UIST '18)*. Association for Computing Machinery, New York, NY, USA, 779–792. https://doi.org/10.1145/3242587.3242605

[40] Huy Viet Le, Sven Mayer, and Niels Henze. 2019. Investigating the Feasibility of Finger Identification on Capacitive Touchscreens using Deep Learning. In *Proceedings of the 24th International Conference on Intelligent User Interfaces* (2019-03-17) *(IUI '19)*. ACM, Marina del Ray, CA, USA. https://doi.org/10.1145/3301275.3302295

[41] Huy Viet Le, Sven Mayer, Benedict Steuerlein, and Niels Henze. 2019. Investigating Unintended Inputs for One-Handed Touch Interaction Beyond the Touchscreen. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services* (Taipei, Taiwan) *(MobileHCI '19)*. Association for Computing Machinery, New York, NY, USA, Article 34, 14 pages. https://doi.org/10.1145/3338286.3340145

[42] Nicolai Marquardt, Johannes Kiemer, David Ledo, Sebastian Boring, and Saul Greenberg. 2011. Designing User-, Hand-, and Handpart-Aware Tabletop Interactions with the TouchID Toolkit. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces* (Kobe, Japan) *(ITS '11)*. Association for Computing Machinery, New York, NY, USA, 21–30. https://doi.org/10.1145/2076354.2076358

[43] Fabrice Matulic, Daniel Vogel, and Raimund Dachselt. 2017. Hand Contact Shape Recognition for Posture-Based Tabletop Widgets and Interaction. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces* (Brighton, United Kingdom) *(ISS '17)*. Association for Computing Machinery, New York, NY, USA, 3–11. https://doi.org/10.1145/3132272.3134126

[44] Sven Mayer, Huy Viet Le, and Niels Henze. 2017. Estimating the Finger Orientation on Capacitive Touchscreens Using Convolutional Neural Networks. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces (ISS '17)*. ACM, Brighton, United Kingdom, 220–229. https://doi.org/10.1145/3132272.3134130

[45] Sven Mayer, Xiangyu Xu, and Chris Harrison. 2021. Super-Resolution Capacitive Touchscreens. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (2021-05-08) *(CHI '21)*. Association for Computing Machinery, New York, New York, USA. https://doi.org/10.1145/3411764.3445704

[46] Sundar Murugappan, Vinayak, Niklas Elmqvist, and Karthik Ramani. 2012. Extended Multitouch: Recovering Touch Posture and Differentiating Users Using a Depth Camera. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge, Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 487–496. https://doi.org/10.1145/2380116.2380177

[47] Ian Oakley, Carina Lindahl, Khanh Le, DoYoung Lee, and MD. Rasel Islam. 2016. The Flat Finger: Exploring Area Touches on Smartwatches. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (Santa Clara, California, USA) *(CHI '16)*. ACM, New York, NY, USA, 4238–4249. https://doi.org/10.1145/2858036.2858179

[48] Markus Oberweger and Vincent Lepetit. 2017. Deepprior++: Improving fast and accurate 3d hand pose estimation. In *Proceedings of the IEEE international conference on computer vision Workshops (ICCVW '17)*. IEEE, 585–594. https://doi.org/10.1109/ICCVW.2017.75

[49] John Kangchun Perng, Brian Fisher, Seth Hollar, and Kristofer SJ Pister. 1999. Acceleration sensing glove (ASG). In *Digest of Papers. Third International Symposium on Wearable Computers (ISWC '99)*. IEEE, 178–180. https://doi.org/10.1109/ISWC.1999.806717

[50] Alan Poston. 2000. Human engineering design data digest. *Washington, DC: Department of Defense Human Factors Engineering Technical Advisory Group* (2000), 61–75.

[51] Gonzalo Ramos, Matthew Boulos, and Ravin Balakrishnan. 2004. Pressure Widgets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vienna, Austria) *(CHI '04)*. ACM, New York, NY, USA, 487–494. https://doi.org/10.1145/985692.985754

[52] Jun Rekimoto. 2002. SmartSkin: An Infrastructure for Freehand Manipulation on Interactive Surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Minneapolis, Minnesota, USA) *(CHI '02)*. Association for Computing Machinery, New York, NY, USA, 113–120. https://doi.org/10.1145/503376.503397

[53] Simon Rogers, John Williamson, Craig Stewart, and Roderick Murray-Smith. 2011. AnglePose: Robust, Precise Capacitive Touch Tracking via 3D Orientation Estimation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. ACM, New York, NY, USA, 2575–2584. https://doi.org/10.1145/1978942.1979318

[54] Anne Roudaut, Eric Lecolinet, and Yves Guiard. 2009. MicroRolls: Expanding Touch-screen Input Vocabulary by Distinguishing Rolls vs. Slides of the Thumb. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA) *(CHI '09)*. ACM, New York, NY, USA, 927–936. https://doi.org/10.1145/1518701.1518843

[55] Dominik Schmidt, Ming Ki Chong, and Hans Gellersen. 2010. HandsDown: Hand-Contour-Based User Identification for Interactive Surfaces. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries* (Reykjavik, Iceland) *(NordiCHI '10)*. Association for Computing Machinery, New York, NY, USA, 432–441. https://doi.org/10.1145/1868914.1868964

[56] Robin Schweigert, Jan Leusmann, Simon Hagenmayer, Maximilian Weiß, Huy Viet Le, Sven Mayer, and Andreas Bulling. 2019. KnuckleTouch: Enabling Knuckle Gestures on Capacitive Touchscreens Using Deep Learning. In *Proceedings of Mensch Und Computer 2019* (Hamburg, Germany) *(MuC'19)*. Association for Computing Machinery, New York, NY, USA, 387–397. https://doi.org/10.1145/3340764.3340767

[57] Valentin Schwind, Lorraine Lin, Massimiliano Di Luca, Sophie Jörg, and James Hillis. 2018. Touch with Foreign Hands: The Effect of Virtual Hand Appearance on Visual-Haptic Integration. In *Proceedings of the 15th ACM Symposium on Applied Perception* (Vancouver, British Columbia, Canada) *(SAP '18)*. Association for Computing Machinery, New York, NY, USA, Article 9, 8 pages. https://doi.org/10.1145/3225153.3225158

[58] Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, Daniel Freedman, Pushmeet Kohli, Eyal Krupka, Andrew Fitzgibbon, and Shahram Izadi. 2015. Accurate, Robust, and Flexible Real-Time Hand Tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3633–3642. https://doi.org/10.1145/2702123.2702179

[59] Yoshiki Takeoka, Takashi Miyaki, and Jun Rekimoto. 2010. Z-Touch: An Infrastructure for 3d Gesture Interaction in the Proximity of Tabletop Surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces* (Saarbrücken, Germany) *(ITS '10)*. Association for Computing Machinery, New York, NY, USA, 91–94. https://doi.org/10.1145/1936652.1936668

[60] Edward Tse, Saul Greenberg, Chia Shen, and Clifton Forlines. 2007. Multimodal Multiplayer Tabletop Gaming. *Comput. Entertain.* 5, 2, Article 12 (April 2007), 12 pages. https://doi.org/10.1145/1279540.1279552

[61] Marin Šarić. 2011. LibHand: A Library for Hand Articulation. http://www.libhand.org/ Version 0.9.

[62] Feng Wang, Xiang Cao, Xiangshi Ren, and Pourang Irani. 2009. Detecting and Leveraging Finger Orientation for Interaction with Direct-touch Surfaces. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology* (Victoria, BC, Canada) *(UIST '09)*. ACM, New York, NY, USA, 23–32. https://doi.org/10.1145/1622176.1622182

[63] David Way and Joseph Paradiso. 2014. A Usability User Study Concerning Free-Hand Microgesture and Wrist-Worn Sensors. In *11th International Conference on Wearable and Implantable Body Sensor Networks*. IEEE, 138–142. https://doi.org/10.1109/BSN.2014.32

[64] Daniel Wigdor, Hrvoje Benko, John Pella, Jarrod Lombardo, and Sarah Williams. 2011. Rock & Rails: Extending Multi-Touch Interactions with Shape Gestures to Enable Precise Spatial Manipulations *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 1581–1590. https://doi.org/10.1145/1978942.1979173

[65] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. 2009. User-Defined Gestures for Surface Computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA) *(CHI '09)*. Association for Computing Machinery, New York, NY, USA, 1083–1092. https://doi.org/10.1145/1518701.1518866

[66] Mike Wu and Ravin Balakrishnan. 2003. Multi-Finger and Whole Hand Gestural Interaction Techniques for Multi-User Tabletop Displays. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology* (Vancouver, Canada) *(UIST '03)*. Association for Computing Machinery, New York, NY, USA, 193–202. https://doi.org/10.1145/964696.964718

[67] Mike Wu, Chia Shen, Kathy Ryall, Clifton Forlines, and Ravin Balakrishnan. 2006. Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces. In *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP '06)*. IEEE, 8 pp. https://doi.org/10.1109/TABLETOP.2006.19

[68] Robert Xiao, Scott Hudson, and Chris Harrison. 2016. CapCam: Enabling Rapid, Ad-Hoc, Position-Tracked Interactions Between Devices. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces* (Niagara Falls, Ontario, Canada) *(ISS '16)*. Association for Computing Machinery, New York, NY, USA, 169–178. https://doi.org/10.1145/2992154.2992182

[69] Robert Xiao, Julia Schwarz, and Chris Harrison. 2015. Estimating 3D Finger Angle on Commodity Touchscreens. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces* (Madeira, Portugal) *(ITS '15)*. ACM, New York, NY, USA, 47–50. https://doi.org/10.1145/2817721.2817737

[70] 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, Article 705, 13 pages. https://doi.org/10.1145/3290605.3300935

[71] Yang Zhang and Chris Harrison. 2015. Tomo: Wearable, Low-Cost Electrical Impedance Tomography for Hand Gesture Recognition. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 167–173. https://doi.org/10.1145/2807442.2807480

[72] Zhensong Zhang, Fengjun Zhang, Hui Chen, Jiasheng Liu, Hongan Wang, and Guozhong Dai. 2014. Left and Right Hand Distinction for Multi-Touch Tabletop Interactions. In *Proceedings of the 19th International Conference on Intelligent User Interfaces* (Haifa, Israel) *(IUI '14)*. Association for Computing Machinery, New York, NY, USA, 47–56. https://doi.org/10.1145/2557500.2557525