WorldPoint: Finger Pointing as a Rapid and Natural Trigger for In-the-Wild Mobile Interactions

DAEHWA KIM, Carnegie Mellon University, USA
VIMAL MOLLYN, Carnegie Mellon University, USA
CHRIS HARRISON, Carnegie Mellon University, USA

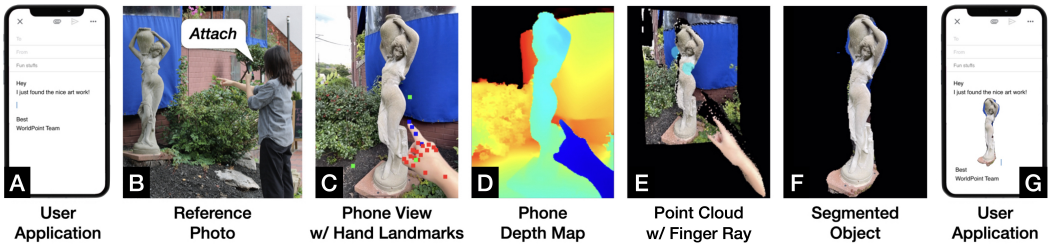

Fig. 1. WorldPoint allows smartphone users to point at objects in the real world for interactive purposes. In this example, a user is writing an email (A). Wishing to attach a photo of a nearby statue, the user points at a statue on the street (B). The phone can see the user's hand and the larger scene in its rear-facing camera (C) and LiDAR depth map (D). The 3D finger pointing vector is extended into the point cloud to mark an object of interest (E), which is segmented from the larger scene (F). When the user says "attach", the pointed object is attached to the email (G). Note the user never has to leave the context of their email, move the phone, or touch the screen, and the whole interaction takes mere seconds. See also Figure 9 for the full UI sequence.

Pointing with one's finger is a natural and rapid way to denote an area or object of interest. It is routinely used in human-human interaction to increase both the speed and accuracy of communication, but it is rarely utilized in human-computer interactions. In this work, we use the recent inclusion of wide-angle, rear-facing smartphone cameras, along with hardware-accelerated machine learning, to enable real-time, infrastructure-free, finger-pointing interactions on today's mobile phones. We envision users raising their hands to point in front of their phones as a "wake gesture". This can then be coupled with a voice command to trigger advanced functionality. For example, while composing an email, a user can point at a document on a table and say "attach". Our interaction technique requires no navigation away from the current app and is both faster and more privacy-preserving than the current method of taking a photo.

CCS Concepts: • **Human-centered computing** → **Graphics input devices**; *Pointing*; *Interaction techniques.*

Additional Key Words and Phrases: Hand pose, augmented reality, mixed reality, XR, pose estimation, image segmentation, mobile devices, computer vision.

ACM Reference Format:
Daehwa Kim, Vimal Mollyn, and Chris Harrison. 2023. WorldPoint: Finger Pointing as a Rapid and Natural Trigger for In-the-Wild Mobile Interactions. *Proc. ACM Hum.-Comput. Interact.* 7, ISS, Article 442 (December 2023), 19 pages. https://doi.org/10.1145/3626478

Authors' addresses: Daehwa Kim, daehwak@cs.cmu.edu, Carnegie Mellon University, Pittsburgh, USA; Vimal Mollyn, vmollyn@andrew.cmu.edu, Carnegie Mellon University, Pittsburgh, USA; Chris Harrison, chris.harrison@cs.cmu.edu, Carnegie Mellon University, Pittsburgh, USA.

2573-0142/2023/12-ART442
https://doi.org/10.1145/3626478

1 INTRODUCTION

Finger pointing is quintessentially human. It is the earliest communicative gesture that develops in infants [18] and all cultures have been found to point with the hands [16]. Pointing is a fast and convenient way to denote an area or object of interest in the real world, which greatly facilitates human-human communication [15]. Unfortunately, in human-computer interactions, pointing is rarely utilized. In the prior work that has utilized pointing interactions (discussed in greater depth later), systems are either room-scale fixed setups (e.g., "Put that There" [11]) or virtual / augmented reality experiences (e.g., [28, 52]). Underexplored, however, is incorporating finger pointing into conventional smartphone interactions. Fortuitously, it is increasingly common for smartphones to feature both wide-angle, rear-facing cameras and hardware to accelerate machine learning models. These are the key technical ingredients to make real-time finger pointing feasible on phones.

More specifically, we envision interactions wherein users utilize smartphone apps as usual, but can raise their hand in front of the phone to point to an item. As one example, a user could be composing an email on their phone, and then point to an object in the environment and say "attach" (Figure 1). Importantly, this interaction never requires leaving the email client or indeed any presses to the touchscreen, and the user is free to continue composition or send the email following the interaction. We believe such lightweight interactions maximize the speed and convenience of finger pointing. Further, by segmenting the object of interest and removing surrounding content, user privacy is enhanced. Figure 2 shows a side-by-side storyboard comparison of WorldPoint compared to iOS 14's equivalent mechanism (see also Video Figure).

In this paper, we describe our proof-of-concept implementation, which runs entirely on a smartphone. Rather than running continuously, which would be too power intensive, we instead periodically run (1 Hz) lightweight models that check only for the binary presence of a hand in front of the smartphone. If a hand is detected, we then run a more intensive model that produces a 3D hand pose at 4 Hz. We then test to see if the user is forming a valid pointing gesture, and if so, we increase tracking FPS to ~20 Hz and we ray cast the finger vector into the scene. We then perform image-based segmentation to "cut out" target objects from the scene. We evaluated our system's pointing accuracy through a user study and found a mean Euclidean error of 15.1 cm, which is roughly half as accurate as innate human pointing accuracy (7.7 cm). Nonetheless, our WorldPoint prototype is still sufficiently accurate to e.g., point to a document lying on a table 2 m away.

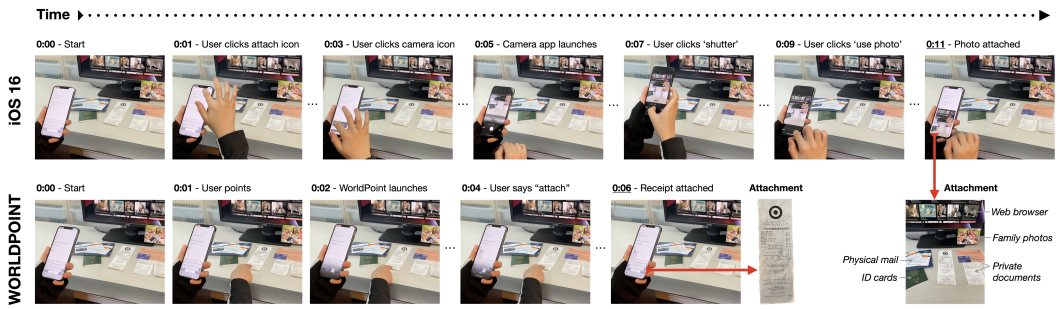

Fig. 2. A real-world comparison of UX flows (WorldPoint vs. iOS 16) for a user attaching a receipt to an email. Note that WorldPoint requires no clicks to the screen and takes 6 seconds, while iOS requires 4 clicks and 11 seconds. Elapsed time is taken from the Video Figure. Ellipsis denotes time gaps >1 second. Note that the iOS attachment contains miscellaneous other content, including potentially private materials. The user could spend additional time better positioning the camera or cropping the photo after capture, further slowing the interaction.

2 COMPARISON TO CONTEMPORARY METHODS & WORLDPOINT DESIGN GOALS

The storyboard in Figure 2 provides one example of how users currently capture or otherwise engage with real-world content in iOS when in an app (which is broadly equivalent to the mechanism in Android 12). In the provided example, a user wishes to attach a paper receipt to an email reimbursement request. In the Gmail app (v6.0.230205) running on iOS 16.3.1, this requires the user to click the attachment icon, then click the camera icon, then take a photo of the item of interest, then confirm by pressing "Use Photo", after which the whole photo is inserted into the email. The interaction takes approximately 11 seconds (see Video Figure). Apple's Mail app also requires 4 clicks and has similar completion times. If the user wished to crop-out surrounding content, multiple additional clicks and swipes would be required. Furthermore, the above interaction sequence takes users away from their application context where the content is desired.

The awkward design of the above interaction motivated several key design goals of WorldPoint: 1) We wished to create an interaction method that did not require navigating away from the current application and losing important context. 2) An intuitive method, closely matching how humans already communicate with one another. 3) A method that was faster than methods found on smartphones today. 4) The targeted object is explicitly selected for unambiguous actions (i.e., only one object is selected, rather than a scene). Finally, 5) the greater scene is obscured so as not to reveal unnecessary information and better preserve user privacy.

Revisiting the interaction sequence in Figure 2, WorldPoint requires no clicks and no navigating away from applications and takes roughly half the time. A combined finger-pointing gesture and voice command allows the user to insert their receipt in-situ, with minimal visual distraction. The elapsed time of both interaction sequences is shown in Figure 2 and the Video Figure.

Two other smartphone pointing methods worth mentioning are pointing the camera at an object, and either taking the center-most object or having the user tap on the desired object with a finger. While imminently possible, such approaches have to be explicitly launched and then occupy the screen. In contrast, WorldPoint can run in the background across all applications, quickly activated when needed. When active, WorldPoint does not take over the screen, preserving task context.

3 RELATED WORK

The sociocultural, cognitive, and biomechanical aspects of human finger-pointing have been extensively studied. A review of this literature is beyond the scope of this work, but we point readers to excellent surveys by Wilson [62], and Butterworth and Jarret [13]. In this section, we focus on human pointing in human-computer interactions, which has been long studied, especially for targeting distant objects (also called distal pointing) [53, 60]. We emphasize this is an extensive literature and a full review is not possible. Instead, we use exemplary systems to convey the research landscape.

3.1 Pointing with Devices

There are innumerable devices that humans have invented to help facilitate or mediate their pointing, from telescoping poles to laser pointers [48, 60]. Digital handheld pointing devices include commercial products such as the Nintendo WiiMote [53, 56] to research devices like "Soap" [10]. Now popular are VR/AR controllers [45, 57, 59] that allow their users to point in 3D virtual space in mid-air. Wearable form factors, including armbands [30], wristbands [25, 36, 37] (including the seminal pointing interaction work "Put-That-There" [11]), rings [64], and even head-mounted cameras [2] have been explored for pointing.

3.2 Finger Pointing on Touchscreens

The success of direct manipulation interfaces was due in no small part to the intuitiveness of pointing (i.e., poking) at 2D interface elements. Since then, researchers have looked at more advanced methods beyond 2D touch input. For instance, Wang et al. [61] enabled more precise 2D selection by leveraging the intersection of two 2D finger rays (using touch ellipsoid to determine finger pitch). Researchers have also looked at ways to do 3D finger pointing on 2D touchscreens, including Xiao et al. [63] and TouchPose [1], which use capacitive image data to estimate the 3D angle of fingers. Holz and Baudisch showed that knowledge of 3D finger angles can be used to increase 2D touch targeting accuracy [32]. More closely related to our work, iOS 16 allows users to long press and drag on an object in a 2D photo, which segments the object. In the accessibility domain, people with visual impairments can explore the pass-through scene on the touch screen by tapping each object [27, 40]. Finally, researchers such as Hürst and van Wezel [34] have explored interacting with on-screen virtual objects in a mobile pass-through AR context.

3.3 Hand Pointing in VR/AR

Although WorldPoint is not a VR/AR method (at no time is the user ever shown a pass-through scene), we note there is considerable research in this space that merits a brief review. Early work by Poupyrev et al. [51, 52] explored finger pointing as a virtual object selection and manipulation technique. Microsoft HoloLens offers a related "Point and Commit" interaction technique [46], though the ray extends from the hand and not a finger. Pointing in VR is also important for human-human interaction in collaborative telepresence applications [42]. Recent advances in computer vision now allow modern VR/AR systems to offer unencumbered 3D hand tracking for interaction [29]. Unique in the literature, however, is an interaction technique for mobile phones combining finger pointing into the real world with voice commands. Further, our implementation is able to run on off-the-shelf smartphone hardware, and is both real-time and infrastructure-free, which is rare. Additional features, such as semantic segmentation of pointed physical objects (and not virtual elements) further differentiates this work from prior systems.

3.4 Hand Pointing in Mobile AR

Closely related to this work are mobile AR finger-pointing interactions. Vincent et al. [58] describe several fundamental methods, including "direct touch" (on the screen) of an AR object of interest, as well as a "screen-centered crosshair", where the user moves the phone such that the object of interest is centered in the field of view of the camera. Even more similar to WorldPoint are methods that utilize the hands in front of the phone to "point" at objects in a real-world scene. For example, Hürst and Wezel [34] used markers attached to the fingers to evaluate translating, scaling, and rotating virtual objects in a mobile AR app where the hands operate in front of the phone, though finger pointing is not implemented. Bai et al. [9] also demonstrated finger tracking in front of a phone, though only 2D tip detection is performed.

In contrast to all of the latter systems, WorldPoint computes a true 3D finger ray cast into a 3D scene. Even more critically, the user experience is very different. All of the prior systems utilize a pass-through AR view that occupies the phone's screen. WorldPoint does not displace current applications, and instead provides a small and segmented preview of only the currently pointed object (i.e., no scene). In fact, we envision users not looking at the screen at all when triggering WorldPoint. Instead, users can look and point at the object of interest, and simply speak a command. Only then does the user need to return to looking at the screen and continue their task. In this way, it is actually more similar to WorldGaze [41], a smartphone interaction technique using a user's gaze ray and spoken commands.

3.5 Finger Pointing Ray Methods

There are two predominant finger-pointing approaches in the literature. First is index finger ray casting (IFRC) [17, 43], also referred to as index finger pointing [39]), pointing ray casting [54], hand-rooted pointing [8] and RayCasting [60]. The second main method is eye-finger ray casting (EFRC) [43, 49], also called eye-rooted pointing [8], head-finger model [26], and gaze finger pointing [39]. Research has shown that different pointing methods have varying offsets and effects (see e.g., [31, 43, 55]). Mayer et al. [43] offer a nice overview of this subject.

WorldPoint could utilize IFRC or EFRC, potentially as a user-toggleable option. However, we chose to use IFRC for our proof-of-concept implementation for three important reasons. First, IFRC better matches how humans point naturally. Second, IFRC is more ergonomic when considering that most users hold their phone at an angle and below the height of their head, which means IFRC can be done in front of the phone without having to lift the hands to near the height of the head for distant targets. Finally, IFRC only requires opening the rear-facing camera (to capture the pointing hand and scene), whereas EFRC additionally requires the user-facing camera to ascertain the 3D position of the user's eyes as well, adding additional processing and power draw.

4 IMPLEMENTATION

We now describe our proof-of-concept implementation. An overview of our pipeline is provided in Figure 3.

4.1 Proof-of-Concept Hardware

We selected an iPhone 12 Pro Max as a proof-of-concept platform. Using its rear-facing camera and LiDAR sensor, this device can provide paired RGB and depth images via Apple's ARKit Dev API at 30 FPS with approximately a 65° field of view. This device also features a 120° ultra-wide angle camera [3], but this is not yet supported by Apple's ARKit API (though indications online suggest this is forthcoming). We also note that while this iPhone contains a rear-facing LiDAR sensor to capture depth data, other LiDAR-less smartphones offer similar depth maps derived from deep learning, SLAM, and other methods (see e.g., Android's equivalent ARCore Raw Depth API [21]).

4.2 Wake Gesture

Like wake words (e.g., "hey Siri", "hey Google"), wake gestures [50] must be sufficiently unique so as not to trigger falsely or by accident. Although finger pointing is natural and common, it is

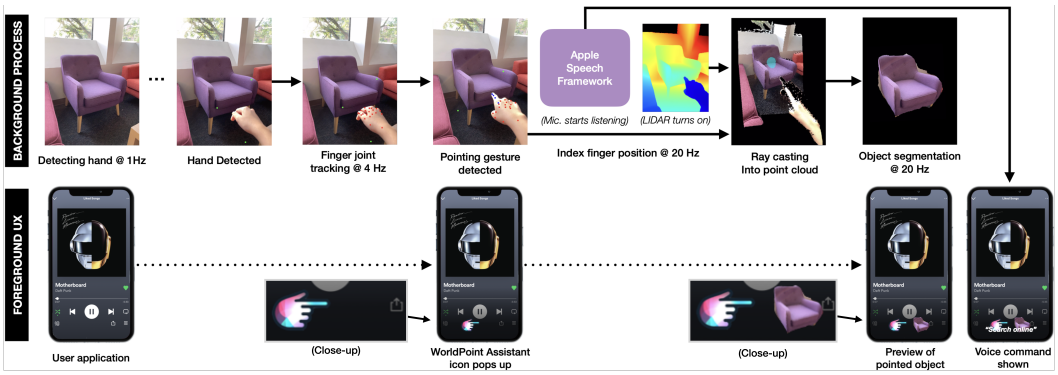

Fig. 3. WorldPoint pipeline with key steps shown in the data pipeline (top) and user interface (bottom).

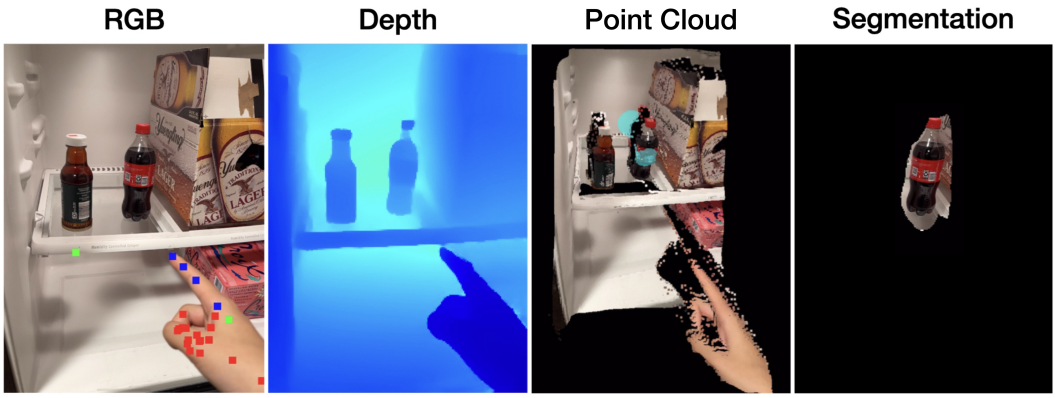

Fig. 4. From left to right: 1) RGB image with corners of the hand bounding box (green dots), land handmarks (red dots), and index finger joints (blue dots) overlaid. 2) Depth Map. 3) Composited point cloud with points intersecting the finger ray highlighted in light blue. 4) Object segmentation result. See also Figure 12 for external reference photos of the user and scene, as well as screenshots of the UI during this interaction.

uncommon for users to perform this gesture in front of their phones at close range, and thus we believe it can serve as a good wake gesture. Example WorldPoint wake gestures can be seen in Figures 1, 9, 10, 12, 11, and 13. This corresponds to the phone held at a comfortable reading distance, with the arm intentionally extended in front of the body as a trigger. This is most comfortable with the arm kept below the shoulder and with the elbow slightly bent. Note this keeps the arm considerably lower, and thus more comfortable, than if we had employed an EFRC pointing method.

4.3 Hand Detection

The first step of our software pipeline is to detect whether a hand is present in front of the smartphone (Figure 3, *Detecting hand @ 1 Hz*). For this, we use MediaPipe's Palm Detector [23] running as a TensorFlow Lite [12] model [24], with a confidence setting of 0.5. To conserve power, we convert 1920×1440 resolution to 256×256 frames and run our model at 1 Hz, sleeping the rest of the time. If a hand candidate is detected, we then examine the bounding box to test if the hand is sufficiently large to be the user's. This eliminates other distant hands in the scene (i.e., from other people) as well as user hands that are held too close or far from the phone. If the hand passes our checks, we move to the next stage of our pipeline.

4.4 Hand Pose & Pointing

With a candidate hand detected, we increase our sampling rate to 4 Hz. We run MediaPipe's Hand Landmark Model [24] (also as a TFLite model) on the candidate bounding box (confidence setting of 0.7; Figure 3, *Index finger position @ 20 Hz*). If a hand pose is generated, we then test to see if it is held in a pointing pose. For this, we use joint angles to test if the index finger is fully extended and the other fingers are angled and tucked in. If the pose passes our check, we continue to the next step of our pipeline. At this point, we also indicate to the user their "wake gesture" has been detected and tracked with a small onscreen icon (see also Example Uses section).

4.5 Finger Ray Casting

With a hand now detected and held in a pointing pose, we now increase our sampling rate to 20 Hz to provide a more responsive user experience. To compute a 3D vector for where the finger

is pointing, we use the index finger's MCP and PIP keypoints as follows the most common hand-rooted method of index finger ray cast (IFRC) [17] (see also Section 3.5 for a discussion on finger pointing methods and why we selected IFRC). We found this joint combination to be the most stable during piloting, though we note that other joints and even other methods are possible, such as regressing on the index finger's point cloud.

Next, in order to ray cast the pointing vector into the scene and have it correctly intersect with scene geometry, we require 3D scene data (i.e., a 2D image is insufficient). For this, we use Apple's ARKit API, which provides paired RGB and depth images (Figure 4, *RGB* and *Depth*). From these sources, we can compute a 3D point cloud in real world units. We use Apple's Metal Framework [5] to parallelize this computation. Once composited, we extend a ray from the index finger into the point cloud scene. As the point cloud is sparse, we identify the point within a specific distance along the ray (Figure 4, *Point Cloud*), rather than requiring an actual collision.

4.6 Image Segmentation

There are several different ways the finger-pointed location in a scene can be utilized, which we elaborate in Example Uses (Section 6). As a proof-of-concept implementation, we use DeepLabV3 [14] trained on 21 classes from Pascal VOC2012 [19]. This model provides masked instance segmentation and runs alongside the rest of our pipeline at 20 FPS on our iPhone 12 Pro Max. For flat rectangular objects, such as receipts and business cards, we take advantage of Apple's built-in Rectangle Detection API [7]. Lastly, we note there are many other techniques for image segmentation, both classical and deep learning based, which are nicely summarized in [47].

4.7 Speech Triggers

To avoid the Midas Touch problem [35], finger pointing is best combined with an independent input modality that acts as a trigger or clutch. For this, we felt spoken commands were a natural compliment (see example sequence in Figure 1). We enumerate many illustrative commands in the subsequent Example Uses section. To implement this functionality, we use Apple Speech Framework [6] to register keywords and phrases, which then trigger event handlers for specific functionality.

4.8 Commercial Practicality

Our implementation demonstrates that the constituent technical components are possible and present on modern smartphones. However, architecturally speaking, WorldPoint would be engineered in a totally different manner by smartphone makers. For example, restrictions in iOS mean that our pipeline must run as a full-blown user application and not a background process. Further limitations in the ARKit API mean that we must open a video stream, as opposed to capturing periodic and low-resolution photos, even when running our hand detection process at 1 Hz. For these and many other architectural reasons, our pipeline currently consumes an estimated 4.2 W when running and tracking a pointing finger. This was calculated using a USB power monitor when the phone was fully charged, and measuring the difference between the phone open to the home screen (1.6 W) vs. running our app (5.8 W). If we assume pointing interactions needing our full pipeline last on average 5 seconds, this would consume 5.83 mWh of power (or 0.041% of the iPhone's 14,130 mWh battery). However, we include these figures for reference only, as we do not believe it is indicative of how such an interaction technique would be implemented or performed in reality.

Despite prototyping limitations, we still aimed to limit power consumption in several ways to better mimic a commercial implementation. First off, our pipeline does not run when the phone screen is off (which is generally most of the day). When the phone is on, we use a series of gated

processes with increasing energy burdens, not only in terms of processing frequency (1 Hz, 4 Hz, and then finally 20 Hz), but also model complexity (from binary hand detection to 3D hand landmarks + object segmentation + voice keywords). We also note that the need for full video steam + LiDAR data would only happen after a finger pointing gesture has been detected. When no finger is pointing, our process only runs once per second, and this check could occur on a high-efficiency co-processor (in much the same way as always-on audio-based wake word detection occurs on modern hardware).

5 EVALUATION

Human pointing accuracy (not mediated by a device) has been found in prior work to have an angular error of around 4-10° [13]. However, finger pointing accuracy using a smartphone held in a user's non-dominant hand has not previously been studied. Furthermore, we could not find any prior experiment that could offer a direct comparison to WorldPoint (i.e., similar range, scale, and task). For this reason, we ran a separate, matched study to establish baseline human pointing performance, which we discuss next. After this, we describe our main study evaluating the accuracy of pointing with our system.

5.1 Unmediated Finger Pointing Accuracy

We recruited 7 participants (mean age 24.3; six identified as female and four as male) to study innate human finger pointing accuracy without a mediating device such as a smartphone. One participant was left-handed, but we asked them to point with their right hand. We affixed a 5mW red laser pointer to participants' index fingertips with electrical tape. The laser was turned on, and participants were allowed to adjust it until they felt the laser dot accurately represented their pointing location. This was the last time participants saw the laser dot in the study. From this point on, participants wore laser safety glasses that blocked the red dot, but not the environment.

Participants stood in front of a wall with a 5 × 3 grid (50 cm interval, 85/135/185 cm from the floor) of targets, seen in Figure 5. Participants completed our procedure at three different distances from the wall, marked by tape on the floor: 80, 160, and 240 cm. Participants were asked to hold a smartphone in their left hand at a typical reading distance. The smartphone ran our study interface,

Fig. 5. Wall with 5×3 grid of targets (50 cm interval) used for measuring accuracy of unmediated finger pointing and pointing via WorldPoint. Participants repeated the study procedure at three distances, marked on the floor.

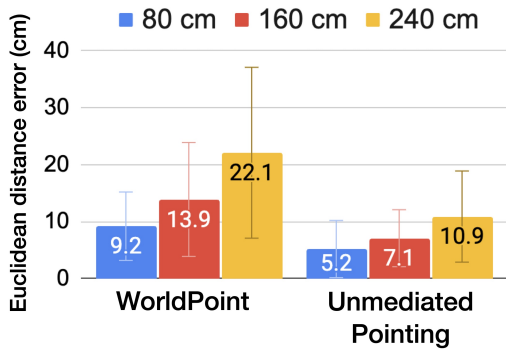

Fig. 6. The Euclidean error distance of unmediated finger pointing and pointing via WorldPoint. Error bars are standard deviations.

which requested wall targets one a time (15 possible targets, 3 repeats, random presentation order, one distance at a time). Participants pointed their finger at the requested target, and when confidence in their pointing direction, announced aloud "next". This prompted the experimenter, on a laptop, to remotely trigger the smartphone to record the requested target, a photo, and a depth map (over a wifi socket). In later analysis, we used these data to manually mark and compute the 3D location of the laser and target (visible in the RGB image, and located in 3D space using the paired depth map) to compute Euclidean error. In total, this procedure produced 945 trails (7 participants $\times$ 15 targets $\times$ 3 repeats $\times$ 3 distances).

5.2 WorldPoint Accuracy

To evaluate pointing accuracy when using WorldPoint, we used an equivalent procedure to our previous study (such that we can directly compare results). Specifically, we use the same 5×3 grid arrangement of targets at the same three distances. No laser was worn this time, and participants held a smartphone running our pipeline in their non-pointing hand. Same as before, participants were told to hold a smartphone at a typical reading distance and were further briefed on how to perform the pointing wake gesture. The study interface ran on the smartphone and showed the next target when the wake gesture was correctly invoked (i.e., after passing our various false positive rejection heuristics).

As before, participants pointed to the target requested on the smartphone study interface, and when satisfied spoke aloud "next". The experimenter, on a laptop, remotely triggered the phone to record several pieces of data computed on the device in real-time: requested target, hand landmarks, pointing vector, RGB image, and depth map. Importantly, the study interface at no time showed participants any visual feedback for correctness (other than if the pointing gesture was invoked or not), nor any live video/images from the camera. This design decision was to prevent any user adaptation to tracking errors.

For this study, we recruited 10 participants (6 of which completed our other study; mean age 22.2; two identified as female and three as male). One user was left-handed, but they were asked to hold the phone in their left hand and point with the right. In total, we captured 1800 pointing trials (10 participants $\times$ 15 targets $\times$ 4 repeats $\times$ 3 distances).

5.3 Results

We first manually inspected all of the study data to codify any errors. Among the 945 trials from the unmediated finger pointing study (i.e., no smartphone involved), we could not locate the laser dot in 35 trials, and these were dropped from the analysis. Either the laser was pointing out of the scene or was too faint to be seen in the captured photo. We also inspected all 1800 trials captured in our WorldPoint accuracy study, finding 30 frames had obvious and gross hand tracking errors (an error rate of 1.7%), which we excluded from our later spatial accuracy analysis. There were also a handful of very errorful outliers which we filtered by dropping any trials greater than 3σ from the mean in each distance condition (35 trials; 1.9% of trials). The latter errors could be detected in real-time and ignored in a fully developed pipeline.

Starting first with unmediated finger pointing accuracy, we found our participants had 16.6, 31.4, 50.1 cm of Euclidean distance error at 80, 160, 240 cm, respectively. Note this is 2D Euclidean distance error in the plane of the wall (where the ray intersects). However, it is known that humans have an offset in where they think they are pointing, and where their finger vector is actually pointing [20, 33, 38, 43, 44]. For this reason, it is important to calculate a per-user offset and correct for this discrepancy. Such a one-time calibration would also have to occur for WorldPoint, perhaps as part of a setup wizard. To compute and apply this post-hoc offset correction, we followed the

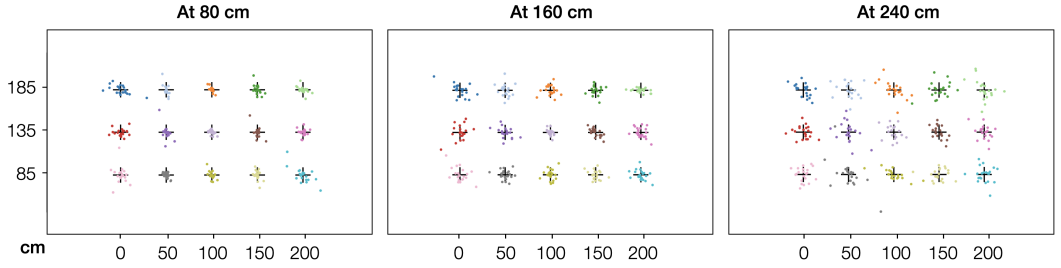

Fig. 7. Scatter plots of unmediated finger pointing trials at 80, 160, and 240 cm distances with per-user offset correction. The fifteen targets are shown as black crosshairs, and each colored dot is a single participant trial. An external reference photo of the setup can be seen in Figure 5.

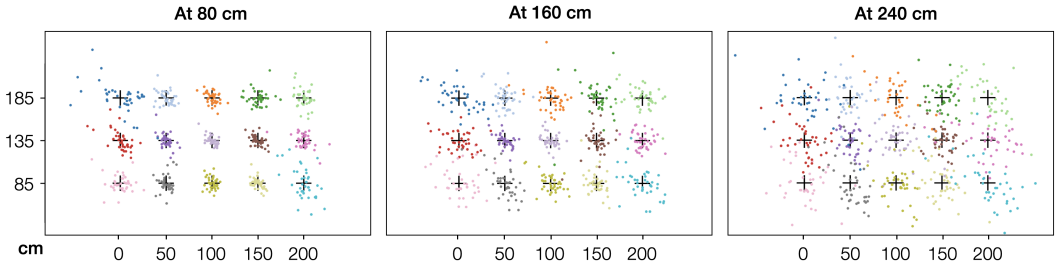

Fig. 8. Scatter plots of all WorldPoint trials at 80, 160, and 240 cm distances with per-user offset correction. The fifteen targets are shown as black crosshairs, and each colored dot is a single participant trial. An external reference photo of the setup can be seen in Figure 5.

procedure used in [32, 65]. Briefly here, this procedure computes the mean Euclidean offset for each user at each distance and subtracts this systemic offset from all of their trials.

After offset correction, unmediated finger pointing Euclidean distance error drops to 5.2 cm (SD = 5.0) at 80 cm, 7.1 cm (SD = 5.0) at 160 cm, and 10.7 cm (SD = 8.0) at 240 cm. With the same one-time offset correction procedure applied to WorldPoint study data, our system achieved a Euclidean distance error of 9.2 cm (SD = 6.0) at 80 cm, 13.9 cm (SD = 10.0) at 160 cm, and 22.1 cm (SD = 15.0) at 240 cm. These results are shown in Figure 6. We also provide scatter plots of unmediated finger pointing and WorldPointing in Figure 7 and 8.

5.4 Contextualizing Performance

Unsurprisingly, our system is not as accurate as unmediated finger input. We hypothesize this is due to a range of factors, including hand landmarking inaccuracies, noise in the depth map, and even extra cognitive load from holding and looking at a smartphone. Fortunately, the first two factors are almost certainly going to improve over time, and our proof-of-concept pipeline should be considered a baseline. Nonetheless, our current WorldPoint pipeline is sufficiently accurate to point to objects roughly 25×25 cm in size, even at a range of 2.4 m. For reference, this is roughly equivalent to a standard-sized piece of paper (US letter or A4). At closer ranges, WorldPoint can target even smaller objects. We believe this is already sufficiently accurate to be useful.

Overall, these results are similar to prior studies investigating index finger ray casting (IFRC). Corradini et al. [17] reported laser pointer's intersection-to-target accuracies of roughly 8.6 cm and 16.2 cm at standing distances of 150 cm and 250 cm, respectively. This is very similar to the unmediated pointing offset we found in our study (7.1 cm at position 160 cm, and 10.7 cm at 240

Fig. 9. In this example sequence, a user is composing an email on their smartphone (A1 & B1). Without exiting the email client, the user points with their finger at a statue they have found (A2). This acts as a "wake gesture" and the phone begins to track what the user is pointing to via its rear-facing camera (C2) and LiDAR, offering a small onscreen preview (B2). Simultaneously, the phone begins to listen for spoken commands. The user says "attach" (A2), which causes the statue to be added to the email (B3). The user then drops their hand (A3) and continues on their way. A1-3: External reference photos; B1-3: Interface screenshots; C1-3: Views from a phone camera, unseen by the user. See also Figure 1 for different steps of the data pipeline.

cm). Lastly, human pointing accuracy not mediated by a device has been found to have an angular error of around 4-10° (see survey in [13]).

6 EXAMPLE USES

We believe the most compelling use of WorldPoint is to augment traditional smartphone applications as a background process, as opposed to taking over the screen with a new interface. In this section, we describe four example categories of use. For each, we built one or more demo applications running our pipeline to help illustrate the potential of the use case (please also see Video Figure). Across all of our demos, we use a Siri-themed pointing hand icon to indicate to the user their wake gesture has been detected (akin to the Siri icon that appears following the "Hey Siri" wake word on iOS devices). We briefly conclude with other ideas we considered, but did not implement.

6.1 "Attach" from World

We created a WorldPoint-augmented email demo with the ability to quickly and conveniently attach images of objects in the real world, such as a document or meal. While composing an email (Figure 9, A1), users simply raise their hand to point to an object (Figure 9, A2). In addition to the WorldPoint icon, a preview of the attachment appears on the screen (Figure 9, B2). If the user wishes to attach an image of this object to their email, they simply say aloud "attach" (Figure 9, A2), without the need for any wake word. This interaction can be repeated in rapid succession for many attachments, or the user can end the interaction by releasing the pointing pose or dropping their hand (Figure 9, A3). Such an attach-from-world interaction need not be limited to an email client and is broadly applicable to any application capable of handling media, including messaging, social media, and note-taking apps.

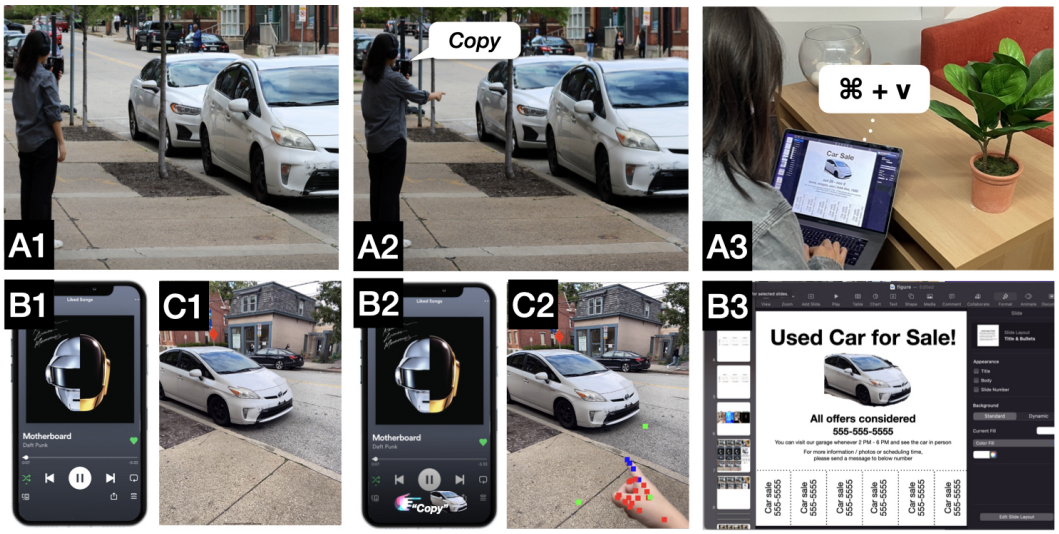

Fig. 10. A user is preparing to sell their car. They go outside to take a photo, listening to their favorite music (1). The user points at the car and says "copy" (2), which copies an image of the car to the iPhone's clipboard. After returning to their office, the user can seamlessly paste it into an advertisement they are designing on their laptop using Apple's Universal Clipboard feature (3). A1-3: External reference photos; B1-3: Interface screenshots; C1-2: Views from a phone camera, unseen by the user.

6.2 "Copy" from World

Whereas our "attach" interaction directs media into the foreground application, it is easy to envision an application-agnostic, system-wide, copy-from-world-to-clipboard interaction. More specifically, at any time, even when not in an application capable of receiving media, the user can point to an object and say "copy". This copies an image of the pointed object to the system clipboard for later use (Figure 10, B2). For this demo, we use iOS' UIPasteboard API. This means Apple's Universal Clipboard (where you can, for example, copy something on your iPhone, but paste it on your Mac) works with our WorldPoint demo (Figure 10, B3).

6.3 "Add to [App]"

WorldPoint could also support more semantically-specific interactions, such as pointing to a business card and saying "add to contacts" (Figure 11, A2) or pointing to a grocery item and saying "add to shopping list" (Figure 12, A2). As before, the latter interactions could happen while the user is in any application (without any need to navigate away from the current task), and the captured information would be passed to the app associated with the spoken command. As a simple demo of this functionality, we simply store captured objects in different lists. We did not, however, write code to parse text (OCR) from captured objects, though obviously this functionality exists.

6.4 Search & Information Retrieval

WorldPoint could also prove useful for search and information retrieval tasks for objects in the world. For instance, a user could be walking down the street scrolling through their Twitter feed, and while passing a restaurant, point to it and say "What's good to eat here?" (Figure 13, A1-3), "what's the rating for this place?" or "what time does this close?" In a similar fashion, a user could point to a car parked on the street and ask "What model is this?" or "How much does this cost?"

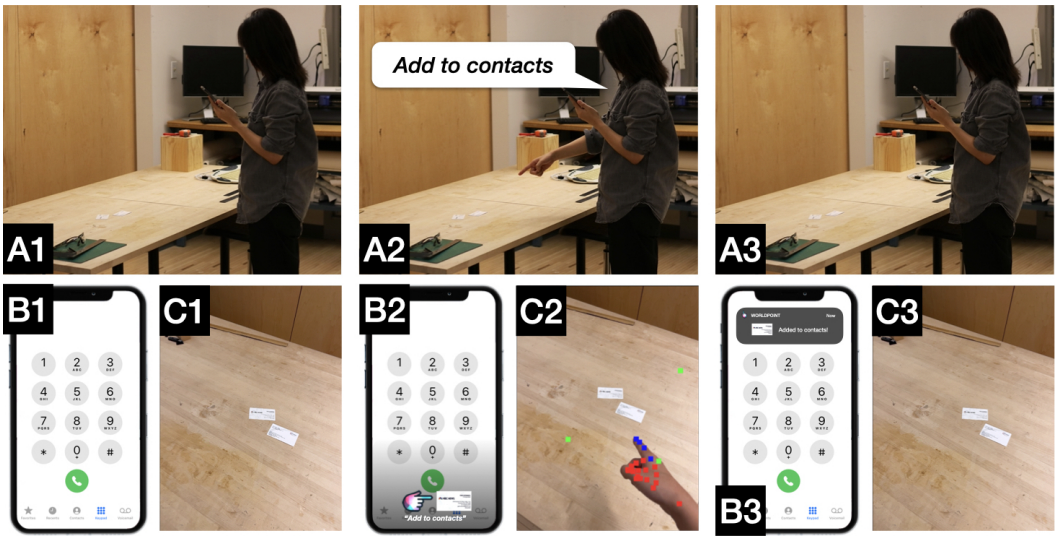

Fig. 11. In this example, a user points to a business card and says "add to contacts". A1-3: External reference photos; B1-3: Interface screenshots; C1-3: Views from a phone camera, unseen by the user.

Fig. 12. In this example, a user points to a beverage and says "add to shopping list". A1-3: External reference photos; B1-3: Interface screenshots; C1-3: Views from a phone camera, unseen by the user (see also Figure 4).

Or, more generally, they could point to an electric scooter and say "Show me more info". These examples were canned demos for illustration (e.g., the restaurant sign was preregistered using Apple's ARKIT ARImageAnchor API [4]), but in practice, WorldPoint would have been tightly coupled to existing services like Google's Voice Assistant and Magic Lens [22].

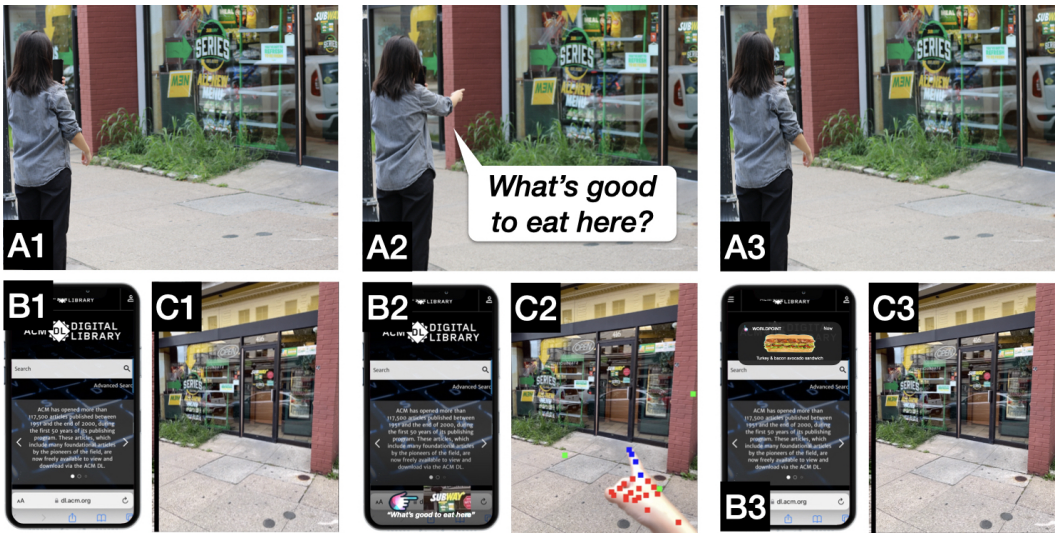

Fig. 13. While surfing the ACM Digital Library, a user passes a new restaurant. To learn more, they simply point at the restaurant and ask "What's good to eat here?". The phone summons relevant information. A1-3: External reference photos; B1-3: Interface screenshots; C1-3: Views from a phone camera, unseen by the user.

6.5 Other Ideas Not Implemented

There were many other interaction ideas for WorldPoint we considered, but did not build into a demo. For example, in human-human interactions, finger pointing can be used to address and issue commands to other humans (e.g., "you go there"), and could likewise work for smart objects (e.g., "on" while pointing at a TV or light switch). Sharing of media is also possible, such as looking at a photo or listening to music on a smartphone, and then pointing to a TV and saying "share", "play here" or similar. It may even be possible to use technologies such as UWB to achieve AirDrop-like file transfer functionality by pointing to a nearby device. Users could also ask questions about the physical properties of objects, such as "How big is this?" or "How far is this?". A drawing app could even eye-dropper colors from the real world using a finger pointing interaction (e.g., "this color").

6.6 Other Pointing & Segmentation Modes

Our pipeline implemented a masking approach to tight segment an object of interest (Figure 14B; see also Section 4.6). This not only brings natural focus to the pointed object but also serves to better preserve privacy (messy bedroom, sensitive documents, other people, etc.). However, this is only one-way finger pointing that could be utilized by end-user applications. For example, and perhaps most straightforward, is to take a small crop centered on the finger point (Figure 14C). This could be done in pixels (e.g., 200 × 200) or real-world units (e.g., 20 × 20 cm). Alternatively, the whole photo could be captured with a "laser dot" (Figure 14D) or "flashlight" effect (Figure 14E) applied to the image to denote a focus point. Another option entirely is for users to perform a lasso action, where they enclose a region in the scene by moving their finger point. The lasso can be terminated by releasing the pointing pose (e.g., transitioning to a fist or open hand gesture) or simply dropping the hand out of view. The lasso could be drawn on the whole image (Figure 14F), or used to "cut out" the lassoed area (Figure 14G).

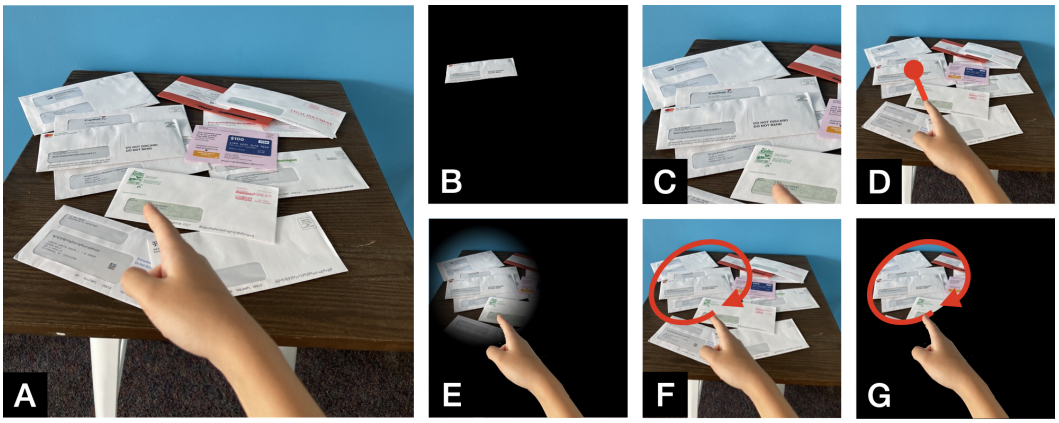

Fig. 14. A user's finger point could support different segmentation styles. A) View from phone camera; B) masked instance segmentation (what we implemented); C) crop of the pointed region; D) "laser dot" rendered onto the full image; E) flashlight effect; F) lassoing on the world image; and G) lasso mask.

7 LIMITATIONS

By running on unmodified, off-the-shelf hardware, WorldPoint must accept some limitations. For instance, Apple's ARKit framework only supports the iPhone's wide-angle lens, but not the ultra-wide lens. This means our pointing gestures have to operate in a smaller volume in front of the phone, even though the device is equipped with a more suitable sensor. Similarly, on iOS, it is not possible to create a background process with intermittent access to the cameras and LiDAR sensor. This, of course, is not an inherent limitation — OEMs such as Apple have the ability to perform such commercial-level integration and optimization.

A more innate limitation is the occasional self-occlusion of a pointing hand captured from an ego-centric viewpoint. Put simply, in some cases we can easily see the hand, but not the finger that is pointing. This most often occurs when the user holds their hand in front of the phone, but the user rotates their wrist to point at something to the non-phone side of their body. This is mitigated in our present pipeline by not registering the hand as a valid wake gesture. However, this can cause frustration among users, who have to try pointing again, potentially adjusting their stance or body direction.

Another limitation is accuracy, both finger pointing and object segmentation. In our user study, we found in a matched task, that human finger pointing was 7.7 cm inaccurate on average. With WorldPoint, our error was 15.1 cm under the same conditions, suggesting our system added roughly 7.4 cm of imprecision. To close this accuracy gap in the future, we believe more advanced hand pose models and higher-resolution sensors will have to be utilized. Nonetheless, our current system is sufficiently accurate for nearby interactions, such as pointing to a document on a table 2 m away, but is insufficient for smaller objects at long ranges.

Additionally, we used an off-the-shelf model, which is reasonably accurate at detecting and masking 21 classes. However, object masking is not perfect, creating graphical artifacts, and the class set is more limited than we might want in practice. Fortunately, deep learning researchers continue to make impressive strides in accuracy and runtime performance, so this capability looks only to improve over time.

We found that our WorldPoint app consumed approximately 4.2 W of power when running our full pipeline at full speed. This is a significant level of power draw and is simply not feasible to

run continuously on a smartphone. However, our full pipeline does not need to run continuously. If pointing interactions last five seconds on average, this equates to 5.83 mWh of power, which has negligible impact on the iPhone 12 Pro Max's 14,130 mWh battery (i.e., each interaction would consume 0.041% of the battery). More important is the background process that periodically checks for the presence of a raised hand in front of the phone. This might run on the order of every second and would have to be very tightly optimized. It would likely have to follow a similar strategy as wake word detection on modern smartphones, where a dedicated co-processor monitors an always-on microphone.

Finally, in terms of validation limitations, we evaluated the pointing accuracy of our pipeline, but not the example interactions we developed. Also, as noted in Evaluation, we did not find sufficiently similar experiments in prior work (distance, device, target, task, etc.) that would provide a direct comparison, so we instead designed a paired study measuring unmediated human pointing accuracy.

8 CONCLUSION

We have presented WorldPoint, which enables finger pointing as a trigger to capture and leverage real-world content. In a commercial implementation, our system could operate in the background, much like voice assistants can today. Also like voice assistants, our interactions do not force the user to switch digital contexts. Our smartphone implementation runs in realtime, leveraging the sensors and hardware-accelerated machine learning found in modern flagship devices. We showed that our prototype system has a pointing error of 15.1 cm across ranges of 0.8 to 2.4 m, which is sufficiently accurate to support a wide range of new interactions.

REFERENCES

[1] Karan Ahuja, Paul Streli, and Christian Holz. 2021. TouchPose: Hand Pose Prediction, Depth Estimation, and Touch Classification from Capacitive Images. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '21)*. Association for Computing Machinery, New York, NY, USA, 997–1009. https://doi.org/10.1145/3472749.3474801

[2] Deepak Akkil and Poika Isokoski. 2016. Accuracy of Interpreting Pointing Gestures in Egocentric View. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Heidelberg, Germany) *(UbiComp '16)*. Association for Computing Machinery, New York, NY, USA, 262–273. https://doi.org/10.1145/2971648.2971687

[3] Apple. 2016. builtInWideAngleCamera API. (2016). https://developer.apple.com/documentation/avfoundation/avcapturedevice/devicetype/2361449-builtinwideanglecamera

[4] Apple. 2022. ARImageAnchor. (2022). https://developer.apple.com/documentation/arkit/arimageanchor

[5] Apple. 2022. Metal Framework. (2022). https://developer.apple.com/documentation/metal

[6] Apple. 2022. Speech Framework. (2022). https://developer.apple.com/documentation/speech

[7] Apple. 2023. Rectangle Detection API. (2023). https://developer.apple.com/documentation/vision/vndetectrectanglesrequest

[8] Ferran Argelaguet, Carlos Andujar, and Ramon Trueba. 2008. Overcoming Eye-Hand Visibility Mismatch in 3D Pointing Selection. In *Proceedings of the 2008 ACM Symposium on Virtual Reality Software and Technology* (Bordeaux, France) *(VRST '08)*. Association for Computing Machinery, New York, NY, USA, 43–46. https://doi.org/10.1145/1450579.1450588

[9] Huidong Bai, Gun A. Lee, and Mark Billinghurst. 2012. Freeze View Touch and Finger Gesture Based Interaction Methods for Handheld Augmented Reality Interfaces. In *Proceedings of the 27th Conference on Image and Vision Computing New Zealand* (Dunedin, New Zealand) *(IVCNZ '12)*. Association for Computing Machinery, New York, NY, USA, 126–131. https://doi.org/10.1145/2425836.2425864

[10] Patrick Baudisch, Mike Sinclair, and Andrew Wilson. 2006. Soap: A Pointing Device That Works in Mid-Air. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology* (Montreux, Switzerland) *(UIST '06)*. Association for Computing Machinery, New York, NY, USA, 43–46. https://doi.org/10.1145/1166253.1166261

[11] Richard A. Bolt. 1980. "Put-That-There": Voice and Gesture at the Graphics Interface. *SIGGRAPH Comput. Graph.* 14, 3 (jul 1980), 262–270. https://doi.org/10.1145/965105.807503

[12] Google Brain. 2022. TensorFlow Lite for mobile and edge. (2022). https://www.tensorflow.org/lite

[13] George Butterworth and Nicholas Jarrett. 1991. What minds have in common is space: Spatial mechanisms serving joint visual attention in infancy. *British Journal of Developmental Psychology* 9, 1 (1991), 55–72. https://doi.org/10.1111/j.2044-835X.1991.tb00862.x

[14] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. 2017. Rethinking Atrous Convolution for Semantic Image Segmentation. *CoRR* abs/1706.05587 (2017). arXiv:1706.05587 http://arxiv.org/abs/1706.05587

[15] S.M. Cooney, N. Brady, and A. McKinney. 2018. Pointing perception is precise. *Cognition* 177 (2018), 226–233. https://doi.org/10.1016/j.cognition.2018.04.021

[16] Kensy Cooperrider, James Slotta, and Rafael Núñez. 2018. The preference for pointing with the hand is not universal. *Cognitive Science* 42, 4 (2018), 1375–1390.

[17] A. Corradini and P.R. Cohen. 2002. Multimodal speech-gesture interface for handfree painting on a virtual paper using partial recurrent neural networks as gesture recognizer. In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, Vol. 3. 2293–2298 vol.3. https://doi.org/10.1109/IJCNN.2002.1007499

[18] Núria Esteve-Gibert and Pilar Prieto. 2014. Infants temporally coordinate gesture-speech combinations before they produce their first words. *Speech Communication* 57 (2014), 301–316. https://doi.org/10.1016/j.specom.2013.06.006

[19] Mark Everingham, SM Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2015. The pascal visual object classes challenge: A retrospective. *International journal of computer vision* 111, 1 (2015), 98–136.

[20] John M Foley and Richard Held. 1972. Visually directed pointing as a function of target distance, direction, and available cues. *Perception & Psychophysics* 12, 3 (1972), 263–268.

[21] Google. 2022. ARCore Depth API. (2022). https://developers.google.com/ar/develop/depth

[22] Google. 2022. Magic Lens. (2022). https://lens.google

[23] Google. 2022. Mediapipe Hands. (2022). https://google.github.io/mediapipe/solutions/hands.html

[24] Google. 2022. Mediapipe Hands TFLite models. (2022). https://google.github.io/mediapipe/solutions/models.html

[25] Boris Gromov, Gabriele Abbate, Luca M. Gambardella, and Alessandro Giusti. 2019. Proximity Human-Robot Interaction Using Pointing Gestures and a Wrist-mounted IMU. In *2019 International Conference on Robotics and Automation (ICRA)*. 8084–8091. https://doi.org/10.1109/ICRA.2019.8794399

[26] Boris Gromov, Jérôme Guzzi, Luca M. Gambardella, and Alessandro Giusti. 2020. Intuitive 3D Control of a Quadrotor in User Proximity with Pointing Gestures. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 5964–5971. https://doi.org/10.1109/ICRA40945.2020.9196654

[27] Anhong Guo, Saige McVea, Xu Wang, Patrick Clary, Ken Goldman, Yang Li, Yu Zhong, and Jeffrey P. Bigham. 2018. Investigating Cursor-Based Interactions to Support Non-Visual Exploration in the Real World. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility* (Galway, Ireland) *(ASSETS '18)*. Association for Computing Machinery, New York, NY, USA, 3–14. https://doi.org/10.1145/3234695.3236339

[28] Carl Gutwin and Saul Greenberg. 2002. A descriptive framework of workspace awareness for real-time groupware. *Computer Supported Cooperative Work (CSCW)* 11, 3 (2002), 411–446.

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

[30] Faizan Haque, Mathieu Nancel, and Daniel Vogel. 2015. Myopoint: Pointing and Clicking Using Forearm Mounted Electromyography and Inertial Motion Sensors. 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, 3653–3656. https://doi.org/10.1145/2702123.2702133

[31] Oliver Herbort, Lisa-Marie Krause, and Wilfried Kunde. 2021. Perspective determines the production and interpretation of pointing gestures. *Psychonomic Bulletin & Review* 28, 2 (2021), 641–648.

[32] Christian Holz and Patrick Baudisch. 2010. The Generalized Perceived Input Point Model and How to Double Touch Accuracy by Extracting Fingerprints. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '10)*. Association for Computing Machinery, New York, NY, USA, 581–590. https://doi.org/10.1145/1753326.1753413

[33] Mengxue Hou, Qiuyang Tao, and Fumin Zhang. 2022. Human Pointing Motion during Interaction with an Autonomous Blimp. (2022).

[34] Wolfgang Hürst and Casper Van Wezel. 2013. Gesture-based interaction via finger tracking for mobile augmented reality. *Multimedia Tools and Applications* 62 (2013), 233–258.

[35] Rob Jacob and Sophie Stellmach. 2016. What You Look at is What You Get: Gaze-Based User Interfaces. *Interactions* 23, 5 (aug 2016), 62–65. https://doi.org/10.1145/2978577

[36] Keiko Katsuragawa, Krzysztof Pietroszek, James R. Wallace, and Edward Lank. 2016. Watchpoint: Freehand Pointing with a Smartwatch in a Ubiquitous Display Environment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces* (Bari, Italy) *(AVI '16)*. Association for Computing Machinery, New York, NY, USA, 128–135.

https://doi.org/10.1145/2909132.2909263
[37] Daniel Kharlamov, Brandon Woodard, Liudmila Tahai, and Krzysztof Pietroszek. 2016. TickTockRay: Smartwatch-Based 3D Pointing for Smartphone-Based Virtual Reality. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology* (Munich, Germany) *(VRST '16)*. Association for Computing Machinery, New York, NY, USA, 365–366. https://doi.org/10.1145/2993369.2996311
[38] Sotaro Kita. 2003. *Pointing: Where language, culture, and cognition meet.* Psychology Press.
[39] Alfred Kranstedt, Andy Lücking, Thies Pfeiffer, Hannes Rieser, and Marc Staudacher. 2006. Measuring and reconstructing pointing in visual contexts. (2006).
[40] Jaewook Lee, Yi-Hao Peng, Jaylin Herskovitz, and Anhong Guo. 2021. Image Explorer: Multi-Layered Touch Exploration to Make Images Accessible. In *Proceedings of the 23rd International ACM SIGACCESS Conference on Computers and Accessibility* (Virtual Event, USA) *(ASSETS '21)*. Association for Computing Machinery, New York, NY, USA, Article 69, 4 pages. https://doi.org/10.1145/3441852.3476548
[41] Sven Mayer, Gierad Laput, and Chris Harrison. 2020. Enhancing Mobile Voice Assistants with WorldGaze. 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, 1–10. https://doi.org/10.1145/3313831.3376479
[42] Sven Mayer, Jens Reinhardt, Robin Schweigert, Brighten Jelke, Valentin Schwind, Katrin Wolf, and Niels Henze. 2020. Improving Humans' Ability to Interpret Deictic Gestures in Virtual Reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376340
[43] Sven Mayer, Valentin Schwind, Robin Schweigert, and Niels Henze. 2018. The Effect of Offset Correction and Cursor on Mid-Air Pointing in Real and Virtual Environments. 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/10.1145/3173574.3174227
[44] Sven Mayer, Katrin Wolf, Stefan Schneegass, and Niels Henze. 2015. Modeling Distant Pointing for Compensating Systematic Displacements. 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, 4165–4168. https://doi.org/10.1145/2702123.2702332
[45] Meta. 2022. Oculus Accessories. (2022). https://store.facebook.com/kr/en/quest/accessories/quest-2/
[46] Microsoft. 2022. Point and commit with hands - mixed reality. https://learn.microsoft.com/en-us/windows/mixed-reality/design/point-and-commit
[47] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2022. Image Segmentation Using Deep Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 7 (2022), 3523–3542. https://doi.org/10.1109/TPAMI.2021.3059968
[48] Dan R. Olsen and Travis Nielsen. 2001. Laser Pointer Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Seattle, Washington, USA) *(CHI '01)*. Association for Computing Machinery, New York, NY, USA, 17–22. https://doi.org/10.1145/365024.365030
[49] Jeffrey S. Pierce, Andrew S. Forsberg, Matthew J. Conway, Seung Hong, Robert C. Zeleznik, and Mark R. Mine. 1997. Image Plane Interaction Techniques in 3D Immersive Environments. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics* (Providence, Rhode Island, USA) *(I3D '97)*. Association for Computing Machinery, New York, NY, USA, 39–ff. https://doi.org/10.1145/253284.253303
[50] Patryk Pomykalski, Mikołaj P. Woźniak, Paweł W. Woźniak, Krzysztof Grudzień, Shengdong Zhao, and Andrzej Romanowski. 2020. Considering Wake Gestures for Smart Assistant Use. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3334480.3383089
[51] Ivan Poupyrev, Mark Billinghurst, Suzanne Weghorst, and Tadao Ichikawa. 1996. The Go-Go Interaction Technique: Non-Linear Mapping for Direct Manipulation in VR. In *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology* (Seattle, Washington, USA) *(UIST '96)*. Association for Computing Machinery, New York, NY, USA, 79–80. https://doi.org/10.1145/237091.237102
[52] Ivan Poupyrev and Tadao Ichikawa. 1999. Manipulating Objects in Virtual Worlds: Categorization and Empirical Evaluation of Interaction Techniques. *Journal of Visual Languages Computing* 10, 1 (1999), 19–35. https://doi.org/10.1006/jvlc.1998.0112
[53] Umar Rashid, Jarmo Kauko, Jonna Häkkilä, and Aaron Quigley. 2011. Proximal and Distal Selection of Widgets: Designing Distributed UI for Mobile Interaction with Large Display. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services* (Stockholm, Sweden) *(MobileHCI '11)*. Association for Computing Machinery, New York, NY, USA, 495–498. https://doi.org/10.1145/2037373.2037446
[54] Robin Schweigert, Valentin Schwind, and Sven Mayer. 2019. EyePointing: A Gaze-Based Selection Technique. In *Proceedings of Mensch Und Computer 2019* (Hamburg, Germany) *(MuC'19)*. Association for Computing Machinery, New

York, NY, USA, 719–723. https://doi.org/10.1145/3340764.3344897

[55] Valentin Schwind, Sven Mayer, Alexandre Comeau-Vermeersch, Robin Schweigert, and Niels Henze. 2018. Up to the Finger Tip: The Effect of Avatars on Mid-Air Pointing Accuracy in Virtual Reality. In *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play* (Melbourne, VIC, Australia) *(CHI PLAY '18)*. Association for Computing Machinery, New York, NY, USA, 477–488. https://doi.org/10.1145/3242671.3242675

[56] Garth Shoemaker, Leah Findlater, Jessica Q. Dawson, and Kellogg S. Booth. 2009. Mid-Air Text Input Techniques for Very Large Wall Displays. In *Proceedings of Graphics Interface 2009* (Kelowna, British Columbia, Canada) *(GI '09)*. Canadian Information Processing Society, CAN, 231–238.

[57] Valve. 2022. Index Controller. (2022). https://www.valvesoftware.com/en/index/controllers

[58] Thomas Vincent, Laurence Nigay, and Takeshi Kurata. 2013. Precise pointing techniques for handheld augmented reality. In *IFIP Conference on Human-Computer Interaction*. Springer, 122–139.

[59] VIVE. 2022. VIVE Tracker 3.0. (2022). https://www.vive.com/us/accessory/tracker3

[60] Daniel Vogel and Ravin Balakrishnan. 2005. Distant Freehand Pointing and Clicking on Very Large, High Resolution Displays. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology* (Seattle, WA, USA) *(UIST '05)*. Association for Computing Machinery, New York, NY, USA, 33–42. https://doi.org/10.1145/1095034.1095041

[61] Feng Wang and Xiangshi Ren. 2009. Empirical Evaluation for Finger Input Properties in Multi-Touch Interaction. 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, 1063–1072. https://doi.org/10.1145/1518701.1518864

[62] F.R. Wilson. 1999. *The Hand: How Its Use Shapes the Brain, Language, and Human Culture*. Knopf Doubleday Publishing Group. https://books.google.com/books?id=VfxvDwAAQBAJ

[63] 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)*. Association for Computing Machinery, New York, NY, USA, 47–50. https://doi.org/10.1145/2817721.2817737

[64] Hui-Shyong Yeo, Juyoung Lee, Hyung-il Kim, Aakar Gupta, Andrea Bianchi, Daniel Vogel, Hideki Koike, Woontack Woo, and Aaron Quigley. 2019. WRIST: Watch-Ring Interaction and Sensing Technique for Wrist Gestures and Macro-Micro Pointing. 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 19, 15 pages. https://doi.org/10.1145/3338286.3340130

[65] Yang Zhang, Junhan Zhou, Gierad Laput, and Chris Harrison. 2016. SkinTrack: Using the Body as an Electrical Waveguide for Continuous Finger Tracking on the Skin. 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, 1491–1503. https://doi.org/10.1145/2858036.2858082

Received 2023-07-01; accepted 2023-09-22