OmniSense: Exploring Novel Input Sensing and Interaction Techniques on Mobile Device with an Omni-Directional Camera

Hui-Shyong Yeo
Huawei
Shenzhen, China
yeo.hui.shyong@huawei.com

Erwin Wu
TokyoTech
Tokyo, Japan
wu.e.aa@m.titech.ac.jp

Daehwa Kim
Carnegie Mellon University
Pittsburgh, PA, USA
daehwak@cs.cmu.edu

Juyoung Lee
KAIST
Daejeon, Republic of Korea
ejuyoung@kaist.ac.kr

Hyung-il Kim
KAIST
Daejeon, Republic of Korea
hyungil@kaist.ac.kr

Seo Young Oh
KAIST
Daejeon, Republic of Korea
seoyoung.oh@kaist.ac.kr

Luna Takagi
TokyoTech
Tokyo, Japan
takagi.l.aa@m.titech.ac.jp

Woontack Woo
KAIST/KI-ITC ARRC
Daejeon, Republic of Korea
wwoo@kaist.ac.kr

Hideki Koike
TokyoTech
Tokyo, Japan
koike@c.titech.ac.jp

Aaron J Quigley
CSIRO's Data61
Sydney, Australia
aquigley@acm.org

Figure 1: A mobile device with an omni-directional camera can capture a wide range of information about the user, including user's hand, body, and the surroundings. This allows various sensing techniques, which enable novel applications for mobile interaction, including danger detection while on the move, body pose for virtual avatar, spatial gestures for gaming, and more.

ABSTRACT

An omni-directional (360°) camera captures the entire viewing sphere surrounding its optical center. Such cameras are growing in use to create highly immersive content and viewing experiences. When such a camera is held by a user, the view includes the user's hand grip, finger, body pose, face, and the surrounding environment, providing a complete understanding of the visual world and context around it. This capability opens up numerous possibilities for rich mobile input sensing. In OmniSense, we explore the broad input design space for mobile devices with a built-in omni-directional camera and broadly categorize them into three sensing pillars: i) near device ii) around device and iii) surrounding device. In addition we explore potential use cases and applications that leverage these sensing capabilities to solve user needs. Following this, we develop a working system to put these concepts into action, by leveraging these sensing capabilities to enable potential use cases and applications. We studied the system in a technical evaluation and a preliminary user study to gain initial feedback and insights. Collectively these techniques illustrate how a single, omni-purpose sensor on a mobile device affords many compelling ways to enable expressive input, while also affording a broad range of novel applications that improve user experience during mobile interaction.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

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

© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-9421-5/23/04…$15.00
https://doi.org/10.1145/3544548.3580747

CCS CONCEPTS

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

KEYWORDS

Omni-directional, 360° camera, input sensing, interaction technique.

ACM Reference Format:
Hui-Shyong Yeo, Erwin Wu, Daehwa Kim, Juyoung Lee, Hyung-il Kim, Seo Young Oh, Luna Takagi, Woontack Woo, Hideki Koike, and Aaron J Quigley. 2023. OmniSense: Exploring Novel Input Sensing and Interaction Techniques on Mobile Device with an Omni-Directional Camera. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), April 23–28, 2023, Hamburg, Germany*. ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3544548.3580747

1 INTRODUCTION

Omni-directional (commonly known as 360°) cameras are used in various ways [31] from photography to capturing experiences. The rising trend of virtual reality (VR) and the increasing desire to capture and view media content in a more immersive and wider format have driven further interest in these cameras. A 360° camera is able to capture not only 2D content, but a 360-degree surrounding content with an omni-directional field of view. This has a number of advantages such as capturing many people in the scene without missing out anyone or any object of interest. It is also much easier for users to capture image and video content in any angle, without requiring them to aim [26, 30] with a viewfinder.

Due to growing demand, mobile 360° camera products are now common in the consumer market. Consumer products such as the Ricoh Theta, Insta360 and GoPro Max are widely available (Figure 2). In parallel to these developments, modern smartphones are increasingly equipped with multiple cameras with wide apertures and viewing angles. Indeed, there are existing smartphones with built-in 360° cameras such as the Acer Holo360 and Protruly series (Figure 2 (bottom)). At present, the more popular hardware manifestation of this relies on a pluggable 360° camera module for a smartphone such as the Insta360 Air, Essential 360 or Huawei Envizion 360 (Figure 2). As such, we can envision a multi-camera software or dedicated 360° camera hardware solution developing in the mobile phone marketplace. The emergence of such cameras presents a platform for novel types of sensing technologies for mobile interaction, which we explore further in this paper.

While typical uses of such cameras focus on capturing image and video content, in this exploration we ask the question: *what kind of input sensing and interaction techniques can be **enabled** if there is an omni-directional camera on a mobile device, such as a smartphone or a tablet?* From the camera's point of view, it can capture the user's hand, finger, body, face, leg, and other objects in the surrounding environment, when the device is held in one's hand (Figure 1 and 3). It can also capture the surface where it is placed, whether on a desk or a charging dock, and activity that happened in the surrounding. Based on these observations, we suggest that omni-directional camera can afford many existing and novel input techniques, using just a single sensor along with standard computer vision techniques. Importantly, with the 360° field of view, it does not require the user to specifically orient the camera to point or focus on specific objects. It works in any orientation.

Figure 2: Example 360° cameras available on the consumer market. From top-left: Ricoh Theta S, Insta360 One X2, GoPro Max 360, Insta360 Air, Huawei Envizion 360, Essential 360 camera module, Acer Holo360, Protruly V10S, D7 and V11S.

These research questions give rise to our name for this approach, namely OmniSense. According to the Oxford dictionary, "Omni" means "of all things", and in "OmniSense" we take an exploratory approach to researching the potential of such sensing techniques to realize a wide range of input capabilities that are afforded by a 360° camera. Our goal is to chart the extent of the design space that such sensing affords us. Overall, we contribute three pillars of input sensing capabilities: i) **near device interaction** (e.g., which hand, which finger, back of the device or 3D fingertip), ii) **around device interaction** (e.g., body pose, hand pose, spatial gesture, proximity or leg), iii) **surrounding device and context-aware sensing** (e.g., environment, human presence, in-car, tabletop or tablet). Such sensing capabilities provide the opportunity to support a range of real-world scenarios and to develop various applications. Following this, we implement a working system, using a built-in 360° camera, to put these concepts into action. We select a large number of promising applications to implement and demonstrate the variety of interactions supported by our solution.

Some, but not all of the ideas presented here have been independently explored in prior work. However, to the best of our knowledge, none of the prior work can enable an all-in-one, omni-purpose sensing with only a single sensor. By contrast, we demonstrate that a single, built-in 360° camera, can achieve all these sensing capabilities and enable novel use cases. These differences emphasize our contributions which go beyond previous work by integrating multiple sensing dimensions to enable an all-in-one, omni-purpose sensing that forms the foundation of OmniSense, in particular, the exploration of broader design space and the technical implementation on off-the-shelf hardware.

Our systematic literature review has highlighted a number of common device usage scenarios and envisioned use cases for on, around and mobile environment interactions. This provides a rich array of interaction patterns, from which we can distill a set of basic interaction primitives. The re-combination of these primitives, in various scenarios, allows us to construct a rich array of new techniques and applications. To explore these techniques and applications, we conduct a technical evaluation and a preliminary user study to gain initial insight and user feedback, which will inform

future research and developments. In broad terms, this paper provides an in-depth exploration of a new class of mobile interaction techniques with omni-directional sensing. Our contributions are therefore multi-fold and can be summarized as follows:

- Comprehensive exploration of using a built-in 360° camera on a mobile device to enable novel input sensing and interaction techniques for mobile interaction, and the exploration of design space, scenarios, user needs and use cases.
- Implementation of a functional, real-time prototype using actual mobile devices with a built-in 360° camera, covering many of the representative sensing capabilities and use cases, demonstrating the technical feasibility of our proposed interactions. In addition, we created video examples for the remaining prototypes.
- A technical evaluation of the sensing results and a preliminary user study to gather initial feedback and insights.
- Our workaround method based on screen capture will enable other researchers to work with 360° cameras even without API access, that is typically not provided by manufacturers.

In summary, we leverage the 360° camera as an omni-purpose sensor for enabling novel input and interaction techniques on mobile devices, by pushing the boundary of what is capable of a single, built-in sensor, going beyond previous work on peripheral sensing.

2 RELATED WORK

Our OmniSense design space is developed from an analysis of prior literature in mobile sensing, input techniques and 360° camera research within the field of HCI. This study of existing work highlights the gaps that exist in the literature, where there is limited availability of solutions that allows continuous sensing of multiple modalities, as shown in comparison Table 1.

Table 1: Comparison with related work. OmniSense supports all sensing capabilities with only a single, built-in sensor.

Methods	Finger (Active, Hover)	Hand (Grip, Pose, Gesture)	Back-Of-Device	Body (Pose, Proximity, Leg)	Environment, Context, Objects	Sensor and Hardware
PreTouch [24]	✓	✓				Custom capacitive sensor
TouchPose [2]	✓	✓				Capacitive touchscreen
Air+Touch [14]	✓	✓				Depth camera
HandSee [74]	✓	✓				Prism attachment
SurroundSee [73]	✓	✓		✓	✓	Panoramic lens attachment
GripSense [16] HandSense [65]		✓				Gyroscope, capacitive sensor
InfiniTouch [37]	✓	✓	✓			Capacitive sensor
PenSight [47]	✓	✓			✓	Wide angle camera
Back-Mirror [66]			✓			Mirror attachment
Finger-Aware Shortcuts [78]	✓					Mirror attachment
Porous Interfaces [20]	✓					Optical sensor on finger
DeepFishEye [51]	✓	✓				Wide angle camera
TapSense [21]	✓					Stethoscope
SideSight [8]	✓					Infrared sensor
SurroundSense [5]					✓	Accelerometer, camera, mic, WiFi
SurfaceSight [36]		✓		✓	✓	360 LiDAR
Project Soli [42]	✓	✓				Radar
BISHARE [79]	✓	✓			✓	Vicon mocap
AirPanes [22]	✓	✓		✓		Vicon mocap
Around Body Interaction [13,14]				✓		Vicon mocap, front camera, IMU
MultiFi [18]		✓		✓		ART outside-in tracking system
MeCap [1]		✓		✓	✓	Spherical mirror attachment
EgoCap [54] Mo2Cap2 [71]		✓		✓		Wide angle camera(s) on headset
Cyclops [11]		✓		✓	✓	Wide angle camera
JackIn Head [32]		✓			✓	Wide angle cameras
Mind The Tap [49]				✓		OptiTrack
Putting Your Best Foot Forward [3]				✓		Accelerometer on leg
Lv et al. [46]				✓		Smartphone rear camera
CrashAlert [23]					✓	Kinect depth camera
WalkSafe [64]					✓	Smartphone rear camera
Director360 [26]				✓	✓	Handheld 360 camera
HindSight [57]					✓	360 camera on helmet
Hand with Sensing Sphere [4]	✓	✓				360 camera on back of hand
OddEyeCam [33]				✓		Wide angle camera
MonoEye [28]				✓	✓	Ultra wide angle camera
BodyTrak [43]				✓		Wide angle camera on hand
Hori et al. [25]				✓		360 camera on hand
OmniSense	✓	✓	✓	✓	✓	360 camera (built-in or add-on)

2.1 Input Sensing on Mobile Device

Improving interaction on mobile devices through novel sensing has been an active research area, where an extensive review can be found in [35]. Here we focus on the five main areas that are highly related to our work, including i) on and above display interaction, ii) back-of-device interaction, iii) around device interaction, iv) around body interaction and v) surrounding and environmental sensing.

i) On and Above Display — First, researchers have augmented input on and above the display to enable more expressive interaction, by recognizing which finger is touching the screen [20, 21, 51, 73, 78], the finger angle [70], the hand pose [2, 51] or the hovering finger [14, 24, 73, 74]. Different sensors such as a capacitive sensor [24, 70], depth camera [14], prism [74] or panoramic lens attachment [73] have been used to enable such capabilities.

ii) Back-of-Device — Researchers also leveraged the space at the back of the device [6, 17, 58] to enable gestures such as back tapping, swiping or scrolling input. Simple tapping can be detected using a built-in accelerometer [27], but other more complex gestures such as swiping or scrolling require adding hardware such as mirror [66], ring [72] or touch sensor [6].

iii) Around Device — A wider space around the device can be useful for sensing how the device is handled by the user [16, 44, 65, 76, 77], the gripping force [53, 61] or where the fingers are [37]. It is also possible to track the non-gripping hand interacting around [22, 42, 79] or beside [8] the device using spatial or surface gestures.

iv) Around Body — The even larger space around the body, including the distance and position of the device relative to the body, can be used for input and interaction [12, 13, 18], such as enabling virtual shelves [39], map navigation [33] or proximity-based screen rotation [73]. Interaction techniques based on leg gestures were also proposed for when both hands are occupied [3, 46, 62].

v) Surrounding and Environment — Finally, sensing the surrounding and environment not only expands the interaction space but also enables context awareness, localization [5] and remote gesture [73]. It is also useful for safety purposes, such as detecting approaching vehicles [40, 64] or obstacles [23, 29] along the path.

Yet, all of the aforementioned methods either require new hardware, or only support a small number of sensing capabilities when using built-in sensors. None is able to offer an all-in-one approach (Table 1). In contrast, OmniSense supports all major sensing capabilities with a single 360° camera already present in smartphones.

2.2 Wide-angle and Omni-directional Camera

Using a wide-angle or omni-directional camera for enabling various interaction techniques has been explored, especially in the HCI and computer vision research communities. For example, researchers have attached a wide-angle camera below a tablet [51] to track fingers, on a stylus [47] to detect hand gestures or on top of a

smartphone [33] for enabling around-body interaction. Researchers have also explored placing a wide-angle camera on the chest [11, 28], headset [54, 71] or on the hand [43] for estimating body posture.

Similarly, 360° cameras are also widely used in research. For example, Jokela et al. [31] conducted a field study on how people use 360° cameras. There is also research work that allows object detection despite the strong distortion [59, 63]. A 360° camera can enable video communication systems with a full spherical display [41] or allows remote users to explore the immersive visual environment of the local user [32]. Further, it can be attached to a cycling helmet [57] and is able to warn cyclists of approaching vehicles outside their field of view. It can be also attached on the hand [4, 25] to enable body-centered interactions. Finally, SurfaceSight [36] enables touch, user, and object sensing using a rotating LiDAR.

Closest to our work, Surround-See [73] uses a 360° panoramic lens attachment (Kogeto Dot) on the smartphone's front camera. It can recognize the device's peripheral environment, objects, and user activities, which facilitates novel use cases for mobile interaction, many of which are also covered in OmniSense's design space. However, the camera lens used in Surround-See has a limited FoV of 56° vertically, which misses significant sensing opportunities, such as the area on the back of the device, the user's face and lower body, and major part of the environment (Figure 8). In this work, we complement these missing capabilities and explore new use cases that are only possible using the omni-directional FoV of 360° camera in a commercial smartphone, along with technical evaluation.

Overall, while there have been independent works that explore using an add-on wide-angle camera or 360° camera for sensing input or objects, we are first to repurpose the built-in 360° camera of a mobile device for sensing the wide extent of the users' hand, finger, body, face and environment to improve mobile interaction.

3 OMNISENSE DESIGN SPACE

The OmniSense design space provides a framework for the consideration of new methods and input techniques, based on the premise that ultra wide-angle or omni-directional cameras will become ubiquitous in future mobile devices. The design space is largely inspired by previous work on mobile sensing, where our analysis indicates several gaps exist in the literature. For example, existing solutions could only achieve limited sensing dimensions, and cannot support sensing of multiple modalities, with most of them requiring custom sensors or an infrastructure tracking system.

Although wide-angle cameras have been explored, there is a large difference between wide-angle and the 360° camera we used. Importantly, a 360° camera can capture more visual information than a common wide-angle camera (Figure 3 (left)), such as the gripping hand, finger interacting near the device, or on the back of the device, the lower body part actions, arm motion and spatial gestures, and etc. Hence, our goal is to explore the breadth of this interconnected design space, going beyond single-point solutions.

OmniSense Input Sensing is a family of sensing capabilities for omni-directional cameras, owing to the uniqueness of such cameras which allows sensing of a broad range of inputs that are broadly categorized into three pillars, i) near the device (contact and proximate), ii) around the device (body, hand, face, leg) and iii) surrounding the device (environment, context, surface). Note, user actions and activities can encompass one or multiple sensing pillars, as shown in Figure 3, where transitioning between them opens up a further range of novel interactions and applications.

OmniSense User Needs and Use Cases is the rationale for the proposed use cases and applications drawn from OmniSense's design space. The interconnected design space affords us the opportunity to explore a wide range of techniques, applications, and scenarios, while considering emerging user needs and issues in using mobile devices. For example, each use case corresponds to solving user needs, using either single or combinations of sensing dimensions, which we illustrate in the following sections. While a use case that includes a single sensing dimension demonstrates the feasibility of a single-point example, a use case that leveraged multiple sensing dimensions unleashes the true potential of *Omni*Sense.

In the following sections, we first discuss what type of sensing capabilities can be achieved, and categorize them into three major pillars. Then, we discuss what applications and scenarios can be enabled and realized with such sensing. Note that some use cases trace through the design space, drawing on multiple sensing dimensions. From this, we then implement interactive prototypes that were designed to explore and characterize a variety of interaction techniques across the proposed design space.

While our focus here is on a smartphone, we suggest these techniques are not limited to just smartphones, but also applicable to other form factors, such as standalone handheld 360° cameras or tablets. In fact, newer standalone 360° cameras are adopting many aspects of a smartphone, e.g., some current generation 360° cameras are running a smartphone OS internally (Ricoh Theta and Acer Holo360 use Android OS). While some are equipped with tiny color touchscreens (GoPro Max, Insta360 X3), which can be difficult to operate due to the fat finger problem.

3.1 Input Sensing Dimensions and Capabilities (What can be sensed?)

By considering interaction techniques through the lens of 360° imagery, what information can the camera capture? As depicted in Figure 3 (left), we can observe that the gripping hand and fingers are always visible when the user holds the device. The non-gripping hand is also visible when the user interacts with the screen using two hands, either touching, hovering, or gesturing above the screen. Owing to the wide field of view, the user's body, face, arm, and legs are also visible most of the time, near the optical center. When the device is resting flat on a surface (e.g., desk) or mounted (e.g., phone charger), the surrounding surface and environment are also clearly visible. Based on these observations, we can postulate that various objects of interest and activities can be tracked with computer vision techniques. Here we discuss these objects in terms of three major pillars, by walking through each example.

*Note: Later in Section 4 (Implementation), we show functional prototypes for most of the representative techniques and applications described here (highlighted in **Bold**). For the remaining, we created video examples and mock-up videos by applying the sensing techniques offline (e.g., post-process with OpenPose). These are underlined.*

3.1.1 ***Near Device Interaction***.

At the near device level - the space proximate to the device contains detailed information about the user's **gripping hand and**

	360° camera view	Sensing dimensions	Use cases	User needs
Near Device	 Hand, Finger	• Handedness [44,65,76] • Grip type / strength [16,37,53] • Which finger [20,51,78] • Fast / slow tap • Pre / post touch [24] • Back-of-device, back-tap [6,17,27,37,58,66] • Hovering finger [14,22,51,73,74]	• Expressive single-handed smartphone interaction: Adaptive UI [16,44,76], PreTouch, BoD [6,17,27,37,58,66], around-body interaction [12,13,18,19,33,39] • Expressive two-handed smartphone interaction: [8] Air+touch [14], PreTouch [24], 3D finger interaction [22,70,73,74,79], seamless mode switching [20,21,78], spatial gesture [4,73], hand pose [2,51], pointing [73] • Tabletop / tablet interaction: Board games, stylus [47,77]	• Situational accessibility (one hand is occupied) • Reachability of single-hand selection • Expressive 3D manipulation input • Natural drawing experience on tablet / tabletop • Visual-occlusion free finger input • Responsive visual updates on screen
Around Device	 Upper Body Full Body	• Body posture (stand / sit) [11] • Body state (walk / run) • Full body skeleton joints [1,28,43,54] • Face feature [1] • Leg gesture [3,46,49,62] • Spatial gesture [4,73] • Hand pose [2] • Orientation / Proximity [12,13,18,33,39,73]	• VR avatar control with full body pose [1,25,28,43] • Leg interaction (pie menu or gesture shortcut) [3,46,49,62] • Body/face touching [45] alert (hygiene monitoring / discreet gesture)	• Immersive video chat experience • Situational accessibility (two hands are occupied) • Breaking unconsciousness/habits
Surroundings	 Left Front Right	• Dangerous cue [23,57,64] • Contextual cue [5,73] • Activity recognition [55,73] • Object detection [36,63,73] • Street, landmark • IoT home devices [73]	• Shoulder surfing prevention • Monitoring attention for keeping screen on • In-car scenario: seat-belt, drowsiness, gesture, etc • On the move (danger detection [23,40,57,64]/social distancing) • Context aware UI [73], intelligent mode switching • Eye-free AR street view without raising phone • Controlling IoT device by gesture / pointing [73] • Meeting, video with full view, active speaker detection [41,55]	• Privacy preserving • Attentive user interface • Lowering cognitive load while driving • Danger detection while walking • Seamless information retrieval of ingredients while cooking • Lowering cognitive load while walking • Rapid IoT device control • Immersive conference call experience

* Each color dot indicates each sensing dimensions (●: near device, ●: around device, ●: surroundings). Three colors dots (●) mean covering all three dimensions.

Figure 3: The OmniSense design space: (*360° camera view*) what the camera can see, from different perspectives such as near, around and surrounding devices. (*Sensing dimensions*) what the system can sense, detect and recognize. (*User needs*) several representative users needs that OmniSense can address. (*Use cases*) what type of use cases and applications can be enabled.

finger. For example, OmniSense can detect the **grip handedness (which hand is holding the device or both hands)** (Figure 1.1 & 1.4) [16] or the grip type (firm or loose). OmniSense can track the **thumb hovering above the screen (Z-height, speed of tap)** (Figure 1.7) [24] or the **fingers on the back-of-device (back-tapping, swiping)** (Figure 1.2) [37, 58]. OmniSense can also track the **3D location of the finger** (Figure 1.5) [74] of the other hand, including the hovering state, pre- and post-touch gesture [14, 24], and speed of tap. OmniSense can also recognize **which part of the finger (index, middle, knuckle) or whether a stylus** (Figure 4d) [21, 73] is being used to touch the screen.

3.1.2 *Around Device Interaction*.

At this level — the space around the device expands the interaction area beyond the screen's boundaries which contains information about the user's body. OmniSense can detect **full body pose** (Figure 1f & 1g) [9], which includes the body posture (standing or sitting) and state (walking or running), **leg gesture** (Figure 1.3) [3], along with head and gaze direction and **facial features (expression, mood)** (Figure 5e). Similar to Surround-See [73], OmniSense can also track the **non-gripping hand's spatial movement around the device** (**gesture, hand pose**, pointing direction) (Figure 1.8) [22, 79] or **touching the face** (Figure 1.9) [45]. Finally, the camera can measure the device's **spatial relationship to the body (proximity, orientation)** (Figure 1.6) [13, 18, 73].

3.1.3 *Surrounding Device and Context Aware Sensing*.

At this level — the extended, large area surrounding the device, including the environment, context and the surface where the device is resting can be sensed, which were explored in Surround-See [73]. For example, when held by a user who is on the move, OmniSense can detect **various objects in front of the users**, such as **approaching cars, humans** and obstacles (Figure 1a). OmniSense can sense the context, such as which room (bedroom or lecture hall) [5, 73] or which vehicle (car or train) the user is currently residing in (Figure 1b). Owing to the omni-directional field of view, it can also recognize landmark buildings (e.g., AR StreetView) or localize indoor positions even when the device is lowered. This does not require the users to raise the device for point-and-shoot, hence lower cognitive load while on the move. When the device is resting on a flat surface (e.g., a desk), or being mounted (e.g., phone charger), it can detect various **objects (e.g., food, tableware, stationery, finger)** (Figure 1d & 1h) or human actions (activity monitoring, in-car gesture) (Figure 1c, 1e & 1i) [73] surrounding the device.

3.2 User Needs and Use Cases

The sensing techniques and methods described thus far operate in an isolated fashion (recognizing which hand, which finger, etc.) without considering use cases. However, our aim is to draw together a holistic suite of interaction techniques across the three pillars of our design space, that can actually address user needs. Hence, in this section, our goal is to explore and enable various applications and scenarios that can improve mobile interaction, in which some scenarios may use one or more pillars, as shown in Figure 3 (*Use cases*). As these are better illustrated in a video, please refer to the **supplementary video** for a demonstration of all the applications.

3.2.1 *Expressive Single and Two-Handed Interaction*.

Currently, there remains several issues on using a mobile device, such as limited one-hand reachability, fat thumb and occlusion issue. Hence, these represent user needs that can be addressed by OmniSense. OmniSense enables following use cases:

i) **Adaptive UI** — Knowing which hand is holding the phone allows for the placement of UI elements intelligently [16]. For example, a menu or virtual keyboard that resizes and adapts to handedness and grip automatically, as shown in Figure 4 (a).

ii) **Ad-lib Interface** — Tracking the thumb's hover state and Z-height allow for an ad-lib interface [24]. For example, Figure 4 (b) demonstrates that the menu interface fades in when the thumb

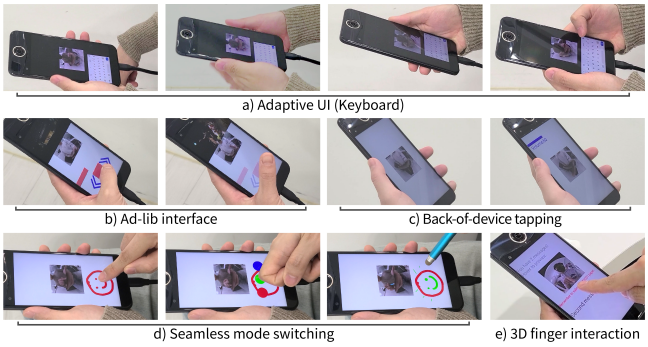


Figure 4: (a) Adaptive UI (keyboard) that adapts to handedness. (b) Ad-lib interface where the menu interface fades in when the thumb approaches. (c) Back-of-device tapping to pull down the notification bar, sliding to adjust the volume. (d) Seamless mode switching, touch from a different finger will switch to different brush, eraser, thickness or trigger menu. (e) 3D finger interaction where finger hovering can preview messages or control scrolling speed.

approaches and fades out when the thumb moves away, so as to not clutter the video content in a full-screen video player.

iii) **Back-of-Device Interaction** — Tracking the index finger on the back-of-device [6] and its 2D location allow for triggering smart actions. For example, tapping on the back of the device acts as shortcut to launch applications, sliding vertically pulls down the notification panel, while sliding horizontally on the back adjusts the volume continuously, as shown in Figure 4 (c).

iv) **Seamless Mode Switching** — Recognizing which finger or object is touching the screen (index, middle, knuckle or stylus) [21, 78] allows for seamless mode switching. For example, in a paint application, touching the screen with the index finger switches to a brush whereas the middle finger switches to an eraser. A touch by the knuckle calls out the color palette, while a stylus changes to a thin brush. All of these work seamlessly without requiring the user to open the menu and choose option, as shown in Figure 4 (d).

v) **3D Finger Interaction** — Tracking the 3D location of the finger above the screen allows for various Air+Touch [14] and Pre-Touch [24] interaction. For example, users can preview a message by hovering their finger on top of it, as shown in Figure 4 (e). A slow tap selects an item normally while a fast tap deletes it. By manipulating the Z-height of the hovering finger, the user can control the scrolling speed of a list.

3.2.2 *Bodily and Spatial Interaction*.

Indeed, a person can be very expressive with the use of gesture, pose and body language to express various intentions. However, these signals are currently not utilized by mobile interaction. OmniSense can capture these body signals to enable expressive bodily and spatial interaction in various applications, such as:

i) **VR Video Avatar Controller** — Tracking the full body pose and facial features allow for immersive interaction and experience (e.g., virtual avatar, virtual YouTuber). For example, in a video call or live streaming, users can control a 3D virtual avatar using full body movement and facial expression, as shown in Figure 5 (a).

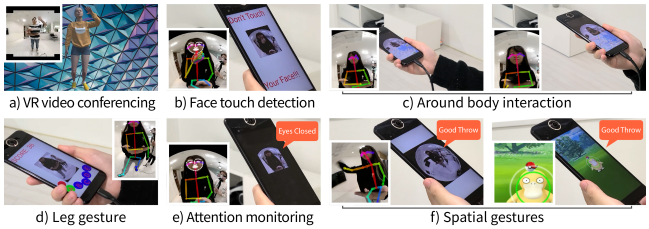


Figure 5: (a) VR avatar control for video conferencing. (b) Face touch detection and warning application. (c) Around body interaction for zooming and panning a map based on the device's proximity and orientation to the user's body. (d) Leg gesture interaction, a dancing game similar to Dance Dance Revolution. (e) Eye closed detection for automatically dimming of screen. (f) Spatial gesture (hand throwing) detection. A good throw of a Poke ball captures the Pokemon!

ii) **Face Touch Detection** — Knowing the hand and body location allows for on-body interaction. For example, the hand touching the body parts or face can be detected [45] to enable mnemonic body shortcut [19] or discreet interaction [38]. It can also give a warning when users unconsciously touch their faces during a pandemic, as shown in Figure 5 (b).

iii) **Around Body Interaction** — Knowing the device's spatial relationship to the user allows for around-body interaction [13, 73]. For example, users can vary the distance of the device to the body for zooming, or alter the position and orientation of the device relative to the body for panning a map, as shown in Figure 5 (c).

iv) **Leg Gesture Interaction** — When both hands are busy or dirty, the legs are free to perform various inputs. Tracking the user's leg allows for leg gesture interaction [3, 46, 49]. A pie menu of multiple selections can be performed by stepping one leg forward, as shown in Figure 5 (d). Precise step counting is also possible.

v) **Stay Awake and Shoulder Surfing Prevention** — Tracking the user's eye gaze allows for intelligently adjusting the screen content. For example, when the user is not looking at the screen, the screen can be dimmed automatically to conserve battery, as shown in Figure 5 (e). If the system detects any stranger other than the owner who is peeking at the screen (e.g., shoulder surfing), the system can automatically hide sensitive information on the screen and warn the user.

vi) **Spatial Gestures** — Tracking the other hand interacting around the device allows for spatial manipulation [22, 79] or pointing [73]. For example, in a game (e.g., Pokemon GO, basketball), a user can perform real hand-throwing gestures rather than swiping on the touchscreen, as shown in Figure 5 (f). Various shortcut commands [47], such as changing the hand sign (C to copy, V to paste, OK to enter) can also be enabled.

3.2.3 *Surrounding and Context-Aware Interaction*.

The user's surroundings and context contain much information for realizing ubiquitous computing. Indeed, prior work such as Surround-See [73] have explored capabilities such as recognizing the device's peripheral environment, objects and user activities in vicinity to the device. OmniSense covers a full 360° FoV that enables further unexplored use cases, such as:

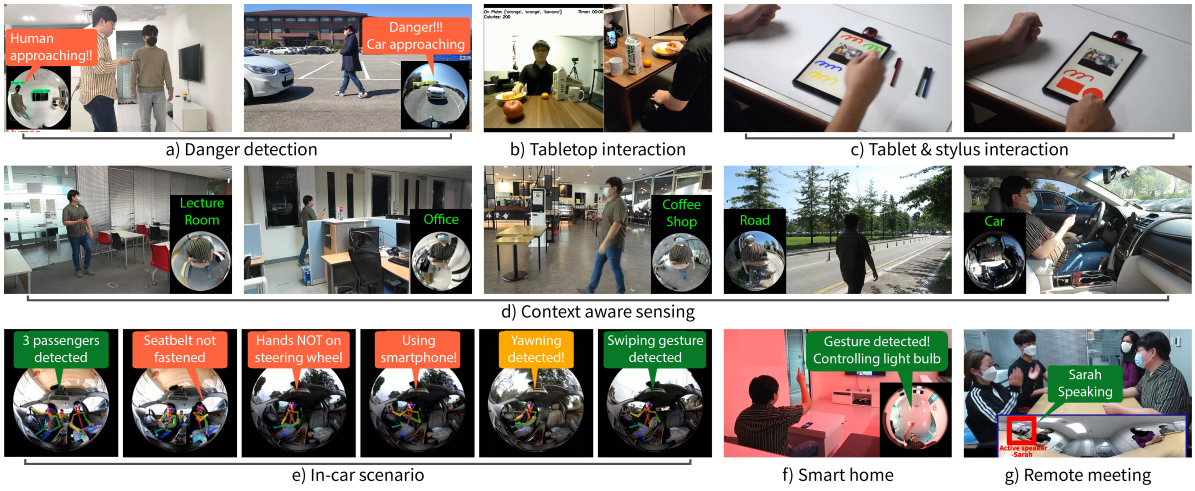


Figure 6: (a) Danger detection and prevention, it warns the user if there are approaching humans or cars. (b) Tabletop interaction: cookpad. (c) Tablet interaction with color stylus and hand pose shortcuts. (d) Context-aware sensing by recognizing the background environment, such as lecture room, coffee shop or in-car. (e) Various events happening inside a car can be detected, including seat occupancy, seat belt reminder, hands-on steering wheel detection, using a smartphone, drowsiness, and hand gestures for infotainment control. (f) Controlling smart home appliances with a simple finger-pointing gesture. (g) In a meeting room, the remote attendee has a full view of the room and all local attendees, plus detection of the active speaker.

i) **Danger Detection** — Understanding the surrounding allows for danger detection and prevention [23, 64]. For example, it is dangerous to keep looking at the phone screen and not paying attention to the road while on the move, commonly known as smartphone zombies. Since the camera has a full view of the environment, such that it can provide a warning to the user if it detects obstacles (an approaching car, pole, or human), as shown in Figure 6 (a). Especially during the pandemic era, it supports practicing social distancing where people are supposed to stay at least two meters away.

ii) **Tabletop, Tablet, and Stylus** — Detecting the objects surrounding the device when it is resting on a surface allows for various tabletop applications. For example, it can enable various board games or tangible interaction using real-world objects [52] for input. In a kitchen scenario, a cookpad application recognizes types of food in a bowl and estimates the calorie amount, while also suggesting recipes. Moving a mug into position sets a timer, which the duration is based on its location, as shown in Figure 6 (b). In addition, it can track the user's finger and allow multi-touch interaction on a large surface [8, 36]. The color of a stylus can be automatically detected for changing the virtual brush's color, while hand pose can be recognized to draw different shapes (fist for circle, palm for square), as shown in Figure 6 (c) [47].

iii) Context Awareness — Understanding the surrounding context also allows for context-aware sensing and interaction. For example, the system can recognize which room (office, cafeteria, bedroom) or which transport vehicle (car, train, flight) the user is currently residing in, and then automatically switches into different modes, as shown in Figure 6 (d). For example, in an office, the phone will switch to silence mode whereas, in a bedroom, it will set an alarm for the next day. In a personal car, it will launch a navigation application whereas in a flight it will switch to flight mode.

iv) In-Car Scenario — Tracking various events happening inside a car is a very compelling use case. This includes detecting the number of passengers and seat occupancy, monitoring driver attention and drowsiness, reminding about unfastened seat belt, warning if hands not handling the steering wheel, detecting hand gestures for controlling infotainment system, detecting child presence, monitoring various vital signs (respiration, heart rate), and finally, capturing fun in-car moments automatically, as shown in Figure 6 (e).

v) Smart Home — When the phone rests on a desk or sofa, it can still track the user's hand and the pointing gesture from a distance. For example, in a smart home scenario, the user can control various appliances just by pointing at them and perform simple gestures [73], while sitting comfortably on the sofa, as shown in Figure 6 (f). It also supports monitoring various human activities, such as if someone entered or leave the room, the user's sitting posture, counting the reps of an exercise, or monitoring how long the user has been using the PC without resting.

vi) Conference Call — In a conference call, it provides a panoramic overview of the entire meeting site [55], recognize all the attendees by name, and detect the active speaker based on detected mouth movement or hand gesture, as shown in Figure 6 (g).

4 IMPLEMENTATION

Translating this broad idea into a practical all-in-one system entails a range of challenges. We created functional prototypes for most of the representative techniques and applications described in the previous section, which are used in the live demo during user study.

Owing to the tremendous advancement in computer vision techniques in recent years, many of the proposed sensing techniques we outlined in this paper can be implemented using mainstream and state-of-the-art (SOTA) techniques, such as convolutional neural

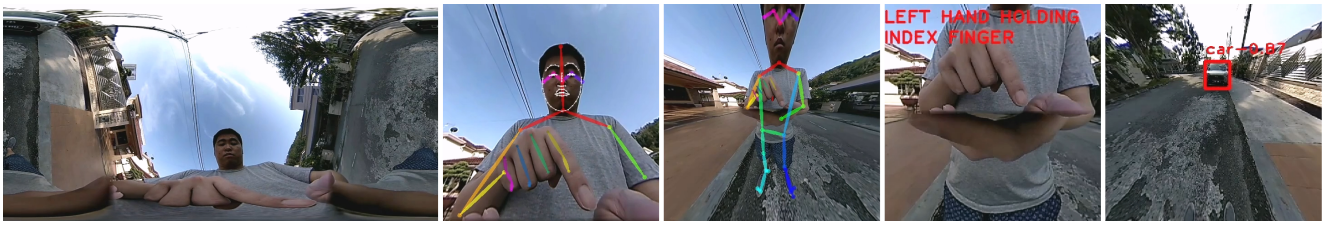


Figure 7: (left) Equirectangular image is converted into multiple perspectives, focusing on different regions of interest, using a corresponding detection module. (middle) body pose & finger (right) approaching the car in the environment is detected.

networks (CNN). Since this paper is focusing on exploration rather than exhaustive implementation and optimization, we consider our implementations as baseline, where future improvement is possible.

4.1 Hardware

Throughout our research, we have investigated a few iterations of hardware prototypes, as shown in Figure 8. Initially, we hot-glue a standalone 360° camera (Ricoh Theta S) on top of a smartphone. This prototype allows high-quality images to be streamed to a PC in real-time using a USB cable but it was uncomfortable to hold.

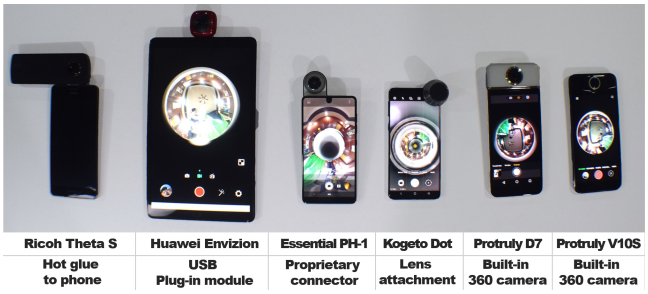


Figure 8: Various hardware we investigated in this work, including (left) camera hot-glued to a smartphone, (middle) pluggable USB camera modules and lens attachment and (right) smartphones with built-in 360° camera.

Next, we explore other, more practical form-factors. The most popular realization of this is the various plug-in 360° camera modules available for smartphones, such as the Huawei and Essential (Figure 8 (middle)). Unfortunately, the manufacturers do not provide API to access the camera image. Next, the low-cost Kogeto Dot lens attachment has a limited field of view of 56° vertically [73] and is not able to see the back of device, face and leg. Therefore, it does not fulfill our requirement for omni-purpose sensing. Lastly, we strive to achieve the most "ready" form-factor, which we employ actual smartphone with a built-in 360° camera, such as the Acer Holo360 or Protruly smartphone. Specifically, the Protruly V10S has a dimension of 16.3 cm x 7.4 cm, as shown in Figure 8 (right). The body is 0.8 cm thick and the thickest part (including protruding lens) is only 1.3 cm. Similarly, there is no API to access the camera image. We mainly use the Protruly V10S smartphone in our data collection, experiment and user study. For tablet use cases, we use the Huawei Envizion 360 pluggable camera module with a tablet, because a tablet with built-in 360° camera does not exist yet.

4.2 Software

In this subsection, we describe software and processing pipeline to unwrap images, capture screens and detect objects.

4.2.1 *Equirectangular Image.*

On the Protruly smartphone, the captured image or video is saved locally in a 360-degree equirectangular format. We convert the equirectangular image to multiple perspective images, each focusing on a different region of interest (ROI) for a different purpose, as shown in Figure 3 and 7. Then, for each ROI, we apply a different detection module. First, for the handedness and finger ROI, we apply four custom detection modules (described in the next Subsection 4.3) which detect handedness, active finger, 2D finger and 3D finger position. For the upper body and full body ROI, we apply the body pose estimation module to extract body joints information, which is further used for inferring body states (e.g., standing vs. sitting, touching the face, eye blinked), as shown in Figure 7 (middle). For the environment ROI, we apply an object detection module to detect approaching obstacles (e.g., car or human for danger detection) or nearby objects (e.g., mug or food for tangible interaction), as shown in Figure 7 (right). While this approach allows tracking bodies and objects in multiple regions simultaneously, it does not allow for a live demonstration, because the captured image is saved in the phone for later access, and there is no API available to access the built-in 360° camera in real-time.

4.2.2 *On-screen Fisheye Image Capture And Overlay.*

We circumvent the limitation of lack of API access with a novel workaround, which is by using a real-time screen capture method with Android debug bridge (ADB). By launching the preloaded stock camera app, the screen renders the camera viewfinder preview in spherical format. Our software then captures the screen content (Figure 9) in real-time and forwards it to a remote PC for image processing and machine learning, using either a USB cable or WiFi.

Field of View — Specifically, the main hardware we used (Protruly V10S smartphone) has a screen resolution of 1920 x 1088 where the spherical preview area is 1088 x 1088 in pixels, with approximately 220+ degree FoV that can be adjusted (drag to rotate viewpoint, pinch to zoom). One downside of this screen capture method is that it is limited to the fisheye view on the screen, rather than the equirectangular format as described previously. Therefore, our processing and recognition pipeline is redesigned to deal with such limitations (e.g., barrel distortion) and built around it.

Opaque Overlay — To demonstrate a prototype system that works in real-time, we need to capture the camera preview on the

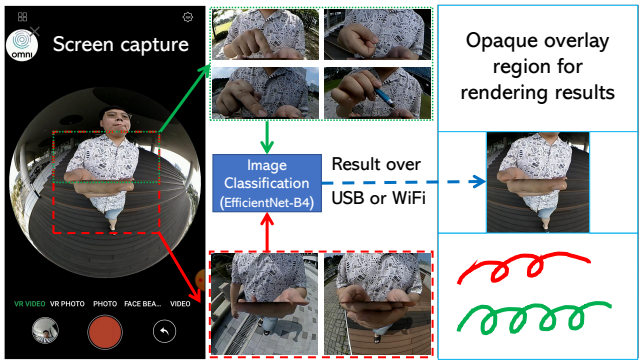


Figure 9: (left) Full-screen capture. (middle) For different use cases, only certain areas are cropped and fed into a neural network classifier. (right) Opaque overlay for rendering the results received from edge device to the user.

screen while also rendering results to the user at the same time. For this reason, we created an opaque overlay using Android service to render results but reserved a transparent region at the center of the spherical area to allow the camera view to be screen captured (Figure 9). No rendering is allowed over this reserved region. We then developed various applications and UI using the overlay. And since the gripping hand and finger are always visible at the center of the image, the transparent region is set to the center 600 x 600 pixels, which only occupied 17% of the total screen's real estate, where the remaining area can be used for rendering, as shown in Figure 9. Furthermore, this cropped region is fed as input to the neural network, which simplifies the machine learning task.

Distortion — Although this fisheye image has barrel distortion, the user's body is close to the center of the image where the distortion is minimal, standard body pose tracking methods will work without modification. In particular, we tried the OpenPose [9] library which worked very well for tracking the body, hand and face, as shown in Figure 10 (a) fisheye.

Un-distortion — Nonetheless, to obtain optimal body pose results, we first unwrap the fisheye image and convert it to different formats (perspective, equirectangular & cubemap), before applying body pose tracking. To calibrate, we took multiple chessboard images from different angles, covering as wide the FoV as possible. Then, we apply calibration using OpenCV and OCamCalib [56] procedure, to extract parameters of the fisheye camera. From the parameters, we generated a pixel-to-pixel conversion map for real-time undistortion, which is applied for every new frame. Examples of the body tracking results for different formats can be seen in Figure 10. Fisheye has the most information, but the body size is unbalanced, where the shoulder appears larger and the legs appear shorter. Cubemap appears to strike a good balance, with minimal information loss (leg slightly cropped when near to edge). In the Perspective image, because the conversion causes high distortion near the edge, we had to limit the FoV (approximately 140°). This causes the leg to be cropped when it is extended. Equirectangular looks good for the middle part of the image, but is highly distorted at the top and bottom part of the image. Further, detailed comparisons are presented in Appendix A.

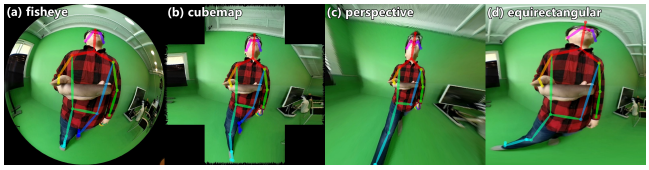


Figure 10: Comparison between different image formats for body pose estimation. From left: fisheye, cubemap, perspective and equirectangular.

4.3 Deep Neural Network Architecture

Since there are no existing models for tracking hands and fingers from the unique camera point of view in our setup, we collect data and train convolutional neural networks from scratch. For the reset, we adopt existing models such as OpenPose [9] for body pose tracking, YOLOv4 [7] for object detection, and I3D [10] for activity recognition. On average, all applications run at an interactive frame rate (> 10 fps) on an Nvidia GTX1080 GPU.

4.3.1 *Handedness and Active Finger Recognition.*

The recognition of handedness and active finger can be posed as a classification task with a few classes. As the hand gripping the phone and the finger touching the screen are always visible in the center of the image, we crop only the center ROI (Figure 9) and fed this as input to a neural network. We adopted the EfficientNet [60] network architecture, as it achieves higher accuracy and efficiency over existing architectures. Specifically, we chose EfficientNet-B4 [60] as it strikes a good balance between accuracy and speed. We replace the last layer with another fully connected layer, using a softmax activation function. Figure 11 shows example dataset.

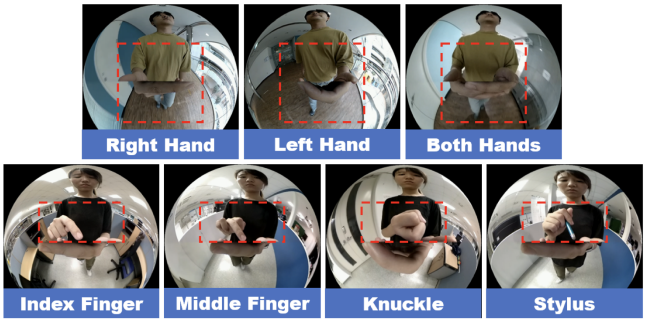


Figure 11: Example dataset for handedness (top) and active finger recognition (bottom). From top-left: right hand, left hand, both hands, index finger, middle finger, knuckle and stylus. Red boxes indicate the cropped region of interest.

4.3.2 *2D and 3D Fingertip Tracking.*

There are existing methods that regress human 3D keypoints from a single image [50, 69, 75], of which the most common ones use heatmap-based regression. Yet, these approaches largely target a single undistorted image. Different from the previous work on body pose estimation, a finger captured by the 360° camera is distorted and occasionally cropped due to its close distance to the camera, thus a conventional method does not work well. Hence, in this

paper, we trained a neural network using the dataset we collected, to achieve faster inference time and obtain higher precision. Our network architecture is shown in Figure 12.

For 2D fingertip tracking, we use a PoseResNet-like [69] encoder-decoder-based convolutional network, which employs ResNet-50 as the backbone to regress 2D heatmaps of different fingertips of both hands. This is similar to the approach used by Xu et al. [71]. Since the 2D regression here is a relatively simple task where the fingertip is clearly shown in the camera view without any occlusion, we suggest a simple yet effective network such as the PoseResNet [69] is sufficient for a baseline implementation, with the advantage of reducing both training and inference time.

For 3D fingertip tracking, estimating the depth of a fingertip from a 2D image is a challenging task, especially for a fisheye lens. Xu et al. [71] succeeded in estimating the distance of body joints from a head-mounted fisheye camera using another stream to estimate the depth map, which is adapted to our case. To achieve better accuracy, we performed a 2-level stacked training to estimate the depth both from the 1st-level output and 2nd-level features, as shown in Figure 12. The network will output a 2D key point heatmap as well as a 3D depth map of the corresponding fingertip which we can extract the 3D camera coordinates from.

Hereby, in the following experiment, we only collected data and trained the network for estimating the three representative joints that are mostly used in mobile interaction: the left thumb, left index, and right index fingertip. Nonetheless, it is possible to expand this tracking to other fingertips or even estimating a full-hand pose.

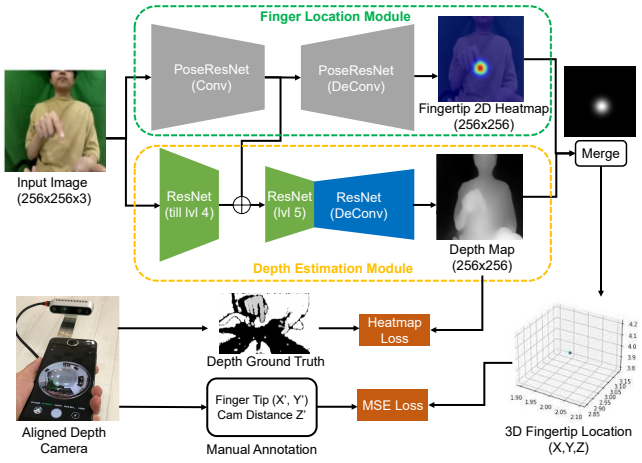


Figure 12: Network architecture for fingertip tracking and data collection setup using depth camera.

5 BASELINE TECHNICAL EVALUATION

We performed a baseline technical evaluation of OmniSense, focusing on detecting hand and finger state, to get a sense of the accuracy and robustness of the system. Since the body pose and object detection results are obtained using state-of-the-art methods from the computer vision literature, we do not find it necessary to evaluate them in this paper. Nonetheless, we examined different un-distortion methods and how well they work with standard body pose estimation model. Please see Appendix A for more details. For each case included in this section, we collect a dataset with our device (Protruly V10S), label them manually, train the respective neural network model, and evaluate the result.

5.1 Handedness and Active Finger Recognition

- Handedness - Which hand is holding the phone? This consists of three classes: left hand, right hand, and both hands.
- Active finger - Which finger is touching the screen? This consists of four classes: index, middle, knuckle and stylus.

5.1.1 *Data Collection.*

We collected data from five participants from a local campus (mean age: 28.6, one female), for two sessions separated by a day. Considering that different users may grip the phone differently, and even a single user may grip the phone differently each time. With this in mind, we designed our data collection procedure to elicit such behaviors. For handedness recognition, participants were asked to hold the phone with their natural and comfortable grip, as if they are using the phone normally. They were asked to vary their handedness and their gripping behavior slightly while walking around the campus (both indoor and outdoor), as shown in Figure 11. For active finger recognition, participants were asked to hold the phone with their left hand, while performing random finger movements on the screen using the right hand's index finger, middle finger, knuckle or a stylus. 1500 images are collected for each case. In total, there are 45000 images for handedness and 60000 images for active finger recognition. We train the EfficientNet-B4 [60] network (with pre-trained weights) with this dataset, for 30 epochs, using a learning rate of 1e-5 and the Adam [34] optimizer.

5.1.2 *Evaluation Procedure.*

We conducted both user-dependent and user-independent evaluations. For the user-dependent test, we take a single day's data for training, and the other day's for testing, then repeat it for 2 folds. For the user-independent test, we take data from both days but leave-1-user-out, train a model for each fold and test on the remaining user data, for 5 folds.

5.1.3 *Results and Discussion.*

For the handedness task, the accuracy is 99.55% (user-dependent) and 98.98% (user-independent). For the active finger task, the accuracy is 97.13% (user-dependent) and 96.12% (user-independent). Overall, the classification results of both the handedness task and active finger task are very accurate and robust, even across users (over 96%). Both of these are rather simple tasks, with well-defined crop area and small input size, owing to the fact that hand and fingers are always visible at the center of image when user is holding and interacting with the touchscreen.

5.2 2D and 3D Fingertip Tracking

- Left index finger touching the back-of-device and hovering above it (2D regression, X, Y location of the fingertip).
- Left thumb touching the front screen and hovering above it (2D regression, X, Y location of the fingertip).
- Right index finger touching the front screen and hovering above it (3D regression, X, Y, Z location of the fingertip).

Table 2: Results of 2D fingertip tracking in real-world and heatmap (HM) errors, and results in 3D and depth (Z-only) errors.

2D Fingertip	General			Leave-1-out		
	MAE (mm)	Heatmap RMSE ($\times 10^{-2}$)	2D-PCK	MAE (mm)	Heatmap RMSE ($\times 10^{-2}$)	2D-PCK
Left Index	2.04	2.16	95.85%	2.78	2.97	92.40%
Left Thumb	2.31	2.00	96.72%	2.72	2.54	93.91%
3D Fingertip	MAE (mm)	Depth MAE (Z only, mm)	3D-PCK	MAE (mm)	Depth MAE (Z only, mm)	3D-PCK
Right Index	4.11	8.25	90.70%	5.46	10.80	89.93%

5.2.1 *Data Collection.*

We collected data from five participants from a local campus (mean age: 28.0, two females). For the left index finger and thumb, we asked the participants to hold the phone with their comfortable grip. They were asked to perform random finger movements while walking around a lab. In total, approximately 49,000 images were collected. 2D (X, Y) location of the left thumb and left index fingertip are annotated manually. For the right index finger, an Intel RealSense depth camera is mounted (Figure 12) on the phone only for collecting ground truth depth data, which is used for the purpose of training the deep neural network. The depth camera is not used during real-time inference and demonstration. Since the depth camera is tethered to a PC, participants remain seated in front of a green backdrop. In total, approximately 30,000 RGB and 30,000 depth images were collected. 3D (X, Y, Z) location of the right index fingertip is annotated by the authors. The 2D heatmap is generated from the XY coordinates using the Gaussian kernel, while the depth map is generated based on the Z-value from a depth camera. All training for the fingertip tracking uses a PyTorch framework, with an initial learning rate of 1e-4, batch size of 64, and Adam [34] optimizer with a multi-step learning rate.

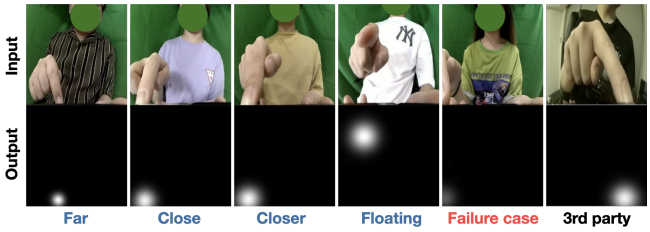


Figure 13: Example results of 3D fingertip estimation. From left: far, close, closer, hovering, failure case and 3rd party (a person not in the dataset). The white point indicates the fingertip position and the size of the points indicates the inverse distance to the camera.

5.2.2 *Evaluation Procedure.*

In both 2D and 3D evaluations, we evaluated a user-dependent general accuracy and a cross-user validation (leave-1-user-out). In the general evaluation, we trained a general model using data from all participants while keeping 20% data randomly from each participant for testing. On the other hand, in the leave-1-user-out evaluation, we trained the network using 4 users' data and evaluated the remaining user, for 5 folds. This aims to study the cross-user generalizability of the model. The results are shown in Table 2.

5.2.3 *Metrics.*

We use a mean absolute error (MAE) as the evaluation metric for the 2D real-world coordinates, of which the unit is in millimeters (mm), similar to related work [51, 67]. To measure the difference in heatmap result, we calculate the root mean square error (RMSE) in pixel-scale to show the similarity of the predicted heatmap and ground truth. We also include a Percentage of Correct Keypoint (PCK) metric with a threshold of 15 mm to represent a more intuitive accuracy. On the other hand, for the 3D fingertip results, as shown in Table 2 (bottom), there are two mean absolute errors (MAE) for either the 3D coordinates or the Z-value only, where the units are all in millimeters (mm). Additionally, the PCK becomes 3D-PCK which is based on the average 3D Euclidean distance between the estimated 3D fingertip position and the ground truth value.

5.2.4 *Results and Discussion.*

For the fingertip tracking, the 2D results have an acceptable accuracy (MAE: Left Index 2.04 mm, Left Thumb 2.31 mm) in the general models and a slightly larger error in cross-user models (MAE: Left Index 2.78 mm, Left Thumb 2.72 mm), where we believe an average error under 3 mm is sufficient for mobile interaction. For comparison, TouchPose [2] used a capacitive touchscreen to reconstruct full hand poses and it showed around 10 mm of error when all fingers are touching the screen (the evaluation setup is different from ours, hence not a direct comparison). Our system tracks the 2D position of only the thumb and index but achieved error of less than 3 mm accordingly. Importantly, the small difference between general and cross-user models indicates the generalizability to new users.

By contrast, the mean errors of coordinates in 3D tracking increase noticeably (MAE: General 4.11 mm, Leave-1-out 5.46 mm). This result is comparable with related work although the settings are not identical. For instance, DeepFishEye [51] tracks five fingertips and yielded errors of ~20 mm for both index and thumb on the smartphone-sized screen. The depth (Z) value has a higher error (MAE of Z-only: General 8.25 mm, Leave-1-out 10.80 mm) when compared to the 2D results. This is because the depth (the distance from the camera) is much more difficult to estimate (with an error of 8 mm~10 mm), especially from a distorted, low-resolution fisheye image. In fact, estimating precise depth information from only a 2D image in real-time remains a challenging task in the CV literature. In addition, the human labeling process may have contributed to the increase in the error rate, especially when the finger is very near to the camera, where a small imprecision in the human annotation may lead to a large error offset. As we used a cropped ROI, when the finger is extended to the extreme, it goes out of view and causes a failure prediction (as shown in Figure 13). In 2D tracking, the prediction jumps to other fingers while in 3D tracking it results in

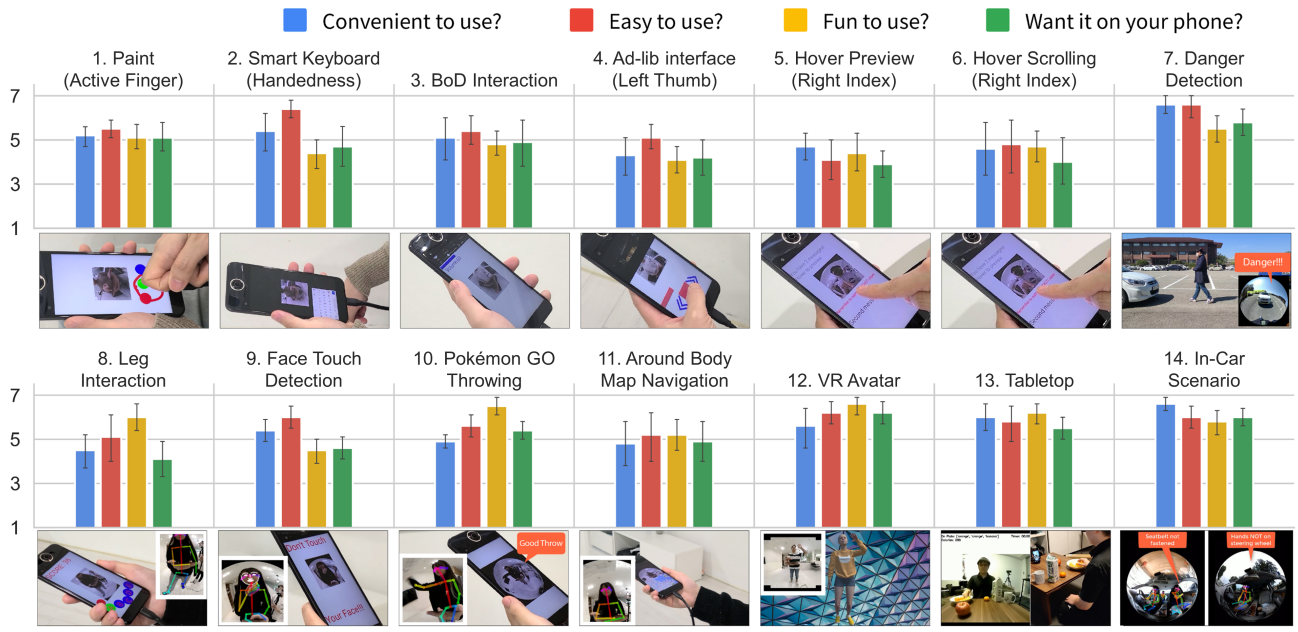


Figure 14: The subjective feedback of participants for each application. 1=Disagree strongly, 7=Agree strongly.

wrong depth estimation. This can be fixed by using a larger ROI, or via filtering. Nevertheless, the overall 3D-PCK is around 90% which means most of the estimated points are close to the ground truth and we suggest that 5 mm mean 3D position error and 10 mm mean depth error is acceptable for many use cases that do not require high precision, such as i) grouping the hover areas based on zones and ii) coarse finger gestures or iii) PreTouch [24] and Air+Touch [14] style of interaction based on hover detection.

6 USER STUDY

To gather feedback and insight about our techniques and applications, we invited participants to try our system. In total, participants tried 13 mini applications (Figure 4-6) spanning the three pillars of the design space. For extra use cases such as the in-car scenario, we created a video example that has actual sensing results overlaid (e.g., OpenPose's tracking of body, face and finger) and show it to participants. Please refer to the main video figure and also **supplementary video** for demonstrations of all applications.

6.1 Participants, Main Task and Protocol

We recruited ten participants from a local University campus (one female, mean age: 28). None of these participants were from the data collection sessions. Among them, nine participants knew about 360° cameras but only five participants have used one before.

We began the user study with an overall introduction, and then an introduction to each of the prototype applications. Participants were given the opportunity to freely try each application for as long as they like, usually lasting one to five minutes for each. After trying out each application, participants completed a questionnaire with four questions on a 7-point Likert scale (Figure 14). Finally, a short interview was conducted to gather feedback and insights from the participants. They were asked about the most and least favorite aspects of their experience, and any other comments they had. The whole study lasted approximately 45 minutes. Participants were given a free drink as a token of appreciation.

6.2 User Study Results and Discussion

Figure 14 shows the subjective feedback of participants regarding the four questions. Results showed that most applications related to the body and surroundings (Demo 7-14) were well received by the participants, but less so for applications related to finger interaction near the device (Demo 1-6).

From the interview, the danger detection (Demo 7) was voted as the most desired application (six participants), followed by 3D list scrolling (two participants). Many participants expressed a high preference for danger detection, the reason being that it can only be enabled with such a unique setup and it is the most "valuable" feature. Two participants commented that they have seen the other applications elsewhere, but danger detection is new and useful to them. They further commented that the accuracy doesn't need to be 100% accurate as there is high potential from this aspect alone.

The VR video avatar (Demo 12) controller application was selected as the most interesting application (six participants), followed by the Pokemon GO (Demo 10) hand-throwing game (two participants' first choice and two participants' second choice). Three participants commented that the VR avatar application is fun. One participant suggested that it is easy to become a VTuber by just holding this phone and another participant commented that it is convenient to just hold the phone within arm's reach as there is no need to setup a camera from afar. Regarding Pokemon GO, participants commented that this real hand-throwing gesture will benefit some games. "*Normally, you cannot play with your body, this is like*

an arcade game which will increase the fun of the game" (P8). Another participant believes that it will be more fun if there are more variations of the throwing gestures (e.g., spinning the Poke ball).

On the other hand, there was a consensus (four participants) that, many of the finger sensing techniques (Demo 1-6) are rather gimmicky. Mainly, they commented that these finger sensing techniques can be enabled by other sensors such as capacitive sensor or radar, which will be more accurate. They see that it is possible to sense the various finger input using 360° camera but they do not think that it is the best solution, considering that other types of sensors may be better. In addition, two participants commented that using menu to switch modes is a well-established method, hence there is no need to change to use a new, gimmicky method to switch modes (Demo 1, which finger), which is more confusing.

Other concerns from participants include robustness of sensing, occlusion problem, battery consumption, lack of usefulness of leg interaction and doubt that smartphones with built-in 360° camera will become mainstream. Surprisingly to us, participants did not complain about the non-fullscreen overlay UI we used to circumvent the camera API limitation. Finally, one of the rewarding comments was: "*It's cool to know that 360° has so many potentials*" (P9), as this is well-aligned with the goal of this exploratory research.

7 OVERALL DISCUSSION

OmniSense broadens our understanding of the types of interactions users can perform on a mobile device with an omni-directional camera. We have implemented functional prototypes that integrate OmniSense's multiple sensing capabilities to enable numerous possible applications and scenarios. From our experience in this integration and from the preliminary user study, we identified open challenges and issues. For example, OmniSense techniques are preferred for body-based and surrounding-based interaction, but less so for the near device interaction (hand and finger). Users found body-based and surrounding-based interaction unique, fun and useful. In contrast, they found near device interaction gimmicky and some said there is no need for such new methods. They also suggest that a customized sensor might be better in terms of robustness and battery consumption (e.g., a capacitance sensor for hover detection). Nonetheless, our exploratory research here is an initial step to uncovering the potential of a single 360° camera for omni-purpose sensing for improving mobile interaction. Optimization and detailed comparison work are left for future work. By exploring the breadth of design space and identifying the usability issues for this new class of mobile interface, we hope that OmniSense can empower users, researchers and manufacturers with an all-in-one sensing solution spanning multiple sensing pillars, for enabling novel applications in the pervasive mobile interaction.

7.1 Is 360° Camera Essential? Potential Solution To Miniaturize 360° Camera Bump

With the increasing FoV of cameras on modern smartphones (120° to 150°), do we require 360° camera for OmniSense? In fact, some use cases that we proposed in this paper can be possible with just a wide-angle camera. On the other hand, some use cases are simply not possible with only 120° to 150° FoV, such as the near device interaction (handedness, active finger and back-of-device). Furthermore, it is not possible to enable multiple use cases simultaneously with small FoV. Thus, we suggest 360° camera is essential to achieve omni-purpose sensing for multiple use cases as outlined here.

As a modern 360° camera uses two wide-angle lenses, there is a stitch line due to the stitching of the two lenses. There is also a tiny blind spot where the camera cannot see, which occurs for objects very close to the center of two lenses. For example, the gripping hand appears to be cut off when it rests on the bezel. Both of these did not cause a major issue in detection, because the neural network will be able to learn and adapt from these.

Perhaps a major concern for the adoption of 360° cameras in modern smartphones is the size. Here, we suggest a potential solution to miniaturize it. In fact, under-display camera technology is becoming matured nowadays. When this technology is combined with a liquid lens that can be morphed on-demand (e.g., Tactus[1]), we envision that it is possible to achieve a smartphone form factor that looks no different than those available now, with a high screen to body ratio and not losing screen space due to fisheye lens.

7.2 Body Pose Estimation Issues, Device Placement And Orientation

For extracting body pose information, we used OpenPose [9] library with a single camera, where the body joints information is 2D only. Hence, it does not support true 3D spatial interaction. Other approaches such as VNect [48] supports real-time 3D human pose estimation with a single RGB camera, which we will experiment with in future work.

In addition, the leg tracking was not robust enough for a smooth leg gesture interaction experience. This is due to two factors: i) occlusion — from the camera's point of view, the hand holding the phone occludes part of the body, causing the lower body to appear as separated from the upper body. ii) shortened legs — as the hand holding the phone is around the chest height, from the camera's point of view, the legs are further away and appear shorter compared to the upper body. Both these factors cause difficulty for robust leg tracking when using the standard OpenPose model, which did not account for such issues. Nonetheless, we found that fisheye distortion is not a major issue, because the body is usually near the center of the image, as shown in Appendix A. We also found that cubemap projection improves the pose tracking slightly in some cases, but there is no clear winner between fisheye or cubemap. Ideally, a solution to improve this issue is to fine-tune the body pose estimation model with images collected from such a high viewpoint (e.g., the annotated dataset from EgoCap [54] or our dataset), which we leave for future work.

Our experiments were conducted where the users hold the phone in front of their body, as if how they normally use a phone. However, there will be occasions where the hand is lowered. While we did not evaluate this condition, recent work by Lim et al. [43] and Hori et al. [25] showed that it is possible to infer body pose using a wide camera on the hand, pointing towards the body. We believe their result is generalizable to our approach.

[1]Tactus Technology Tactile Touchscreen: https://www.youtube.com/watch?v=JelhR2iPuw0

7.3 Privacy Concern

The use of such an omni-directional camera presents a number of privacy concerns. Such concerns have been addressed when cameras have been incorporated elsewhere. For example, some device manufacturers provide a physical kill-switch for a laptop camera. Similarly, this physical kill-switch approach can be adopted for a smartphone 360° camera, which the users can turn off when they do not need OmniSense. Manufacturers also use on-device machine learning inference and on-device processing, where the data are destroyed without ever leaving the device. This can provide users with verifiable service guarantees to offer them confidence in the use of the camera as an on-board only sensor. An alternative approach is to employ a low-resolution sensor, such as infrared, thermal or LiDAR cameras, where it is not possible to reconstruct details of the user's face and identity. For example, Clarkson [15] captures a 360° sphere around the user for 100 days, where he intentionally uses a low-resolution camera with a fisheye lens to avoid privacy issues. Indeed, he shows that faces are too blurry to identify, yet it is possible to successfully recognize a lot of context with the low-resolution images. Hence, privacy with the use of a camera sensor can be addressed but it remains a constant challenge.

8 LIMITATIONS AND FUTURE WORK

In our implementation, real-time processing is performed on a desktop PC, and the result is sent back to the mobile device for displaying to the user. This is largely due to two reasons: i) lack of API to access raw camera images and ii) limited computing resources on mobile devices. Future hardware improvements, system optimizations, and gaining API access to the full 360° camera image will ensure OmniSense can run on the mobile device without requiring offloading to edge devices.

While a custom, single-purpose sensor may be more suitable for independent tasks such as finger hovering or back-tap, it can be costly to add a custom sensor for every desired input modality. In contrast, we demonstrated that with the captured 360° video (equirectangular format), we can enable ***all*** sensing pillars simultaneously, including hand, finger, full body and the surrounding environment. However, for the live demonstration, the lack of camera API access means that we are limited to the screen capture method, which has an approximately 220° FoV. While this 220° FoV can cover multiple use cases simultaneously, such as near device + bodily interaction (two sensing pillars), or upper body + front danger detection + environment sensing (three sensing pillars), it could ***not*** enable ***all*** sensing simultaneously. This problem will be solved by gaining API access to the raw image.

Furthermore, there are a wide range of potential applications that can be enabled by OmniSense that we have yet to explore. So far, we have only implemented thirteen representative applications, which barely scratched the surface of all the possibilities. Other applications such as face/gait authentication, vital signs monitoring [68], indoor positioning, and autonomous driving are possible but require further research and development. We will explore such application areas in future work. While we have demonstrated the potential of such 360° camera on a mobile device, more engineering efforts are required to make it practical for real-world usage, with concerns about battery usage and processing speed.

9 CONCLUSION

We have presented OmniSense, a design space covering a broad range of input sensing and interaction techniques for mobile devices, enabled by a single, built-in 360° camera. Furthermore, our prototypes demonstrated how this single sensor is capable of enabling various compelling use cases and applications. We conducted a baseline evaluation and preliminary study. We hope this exploratory research showcased the potential of 360° camera for enabling novel interaction techniques for improving mobile interaction. And along with the workaround method that we provided, we hope these contributions to knowledge inspire future research on this topic.

ACKNOWLEDGMENTS

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-01270, WISE AR UI/UX Platform Development for Smartglasses), and by Japan Science and Technology Agency (JST) CREST Grant No. JPMJCR17A3.

REFERENCES

[1] Karan Ahuja, Chris Harrison, Mayank Goel, and Robert Xiao. 2019. MeCap: Whole-Body Digitization for Low-Cost VR/AR Headsets. 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, 453–462. https://doi.org/10.1145/3332165.3347889

[2] 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

[3] Jason Alexander, Teng Han, William Judd, Pourang Irani, and Sriram Subramanian. 2012. Putting Your Best Foot Forward: Investigating Real-World Mappings for Foot-Based Gestures. 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, 1229–1238. https://doi.org/10.1145/2207676.2208575

[4] Riku Arakawa, Azumi Maekawa, Zendai Kashino, and Masahiko Inami. 2020. Hand with Sensing Sphere: Body-Centered Spatial Interactions with a Hand-Worn Spherical Camera. In *Symposium on Spatial User Interaction* (Virtual Event, Canada) *(SUI '20)*. Association for Computing Machinery, New York, NY, USA, Article 1, 10 pages. https://doi.org/10.1145/3385959.3418450

[5] Martin Azizyan, Ionut Constandache, and Romit Roy Choudhury. 2009. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking* (Beijing, China) *(MobiCom '09)*. Association for Computing Machinery, New York, NY, USA, 261–272. https://doi.org/10.1145/1614320.1614350

[6] Patrick Baudisch and Gerry Chu. 2009. Back-of-Device Interaction Allows Creating Very Small Touch Devices. 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, 1923–1932. https://doi.org/10.1145/1518701.1518995

[7] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv:2004.10934 [cs.CV]

[8] Alex Butler, Shahram Izadi, and Steve Hodges. 2008. SideSight: Multi-"touch" Interaction around Small Devices. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (Monterey, CA, USA) *(UIST '08)*. Association for Computing Machinery, New York, NY, USA, 201–204. https://doi.org/10.1145/1449715.1449746

[9] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. 2019. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019), 1–1. https://doi.org/10.1109/TPAMI.2019.2929257

[10] J. Carreira and A. Zisserman. 2017. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4724–4733. https://doi.org/10.1109/CVPR.2017.502

[11] Liwei Chan, Chi-Hao Hsieh, Yi-Ling Chen, Shuo Yang, Da-Yuan Huang, Rong-Hao Liang, and Bing-Yu Chen. 2015. Cyclops: Wearable and Single-Piece Full-Body Gesture Input Devices. 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, 3001–3009. https://doi.org/10.1145/2702123.2702464

[12] Xiang "Anthony" Chen, Nicolai Marquardt, Anthony Tang, Sebastian Boring, and Saul Greenberg. 2012. Extending a Mobile Device's Interaction Space through Body-Centric Interaction. In *Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services* (San Francisco, California, USA) *(MobileHCI '12)*. Association for Computing Machinery, New York, NY, USA, 151–160. https://doi.org/10.1145/2371574.2371599

[13] Xiang "Anthony" Chen, Julia Schwarz, Chris Harrison, Jennifer Mankoff, and Scott Hudson. 2014. Around-Body Interaction: Sensing & Interaction Techniques for Proprioception-Enhanced Input with Mobile Devices. In *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices & Services* (Toronto, ON, Canada) *(MobileHCI '14)*. Association for Computing Machinery, New York, NY, USA, 287–290. https://doi.org/10.1145/2628363.2628402

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

[15] Brian Patrick Clarkson. 2002. *Life patterns: structure from wearable sensors*. Ph.D. Dissertation. Massachusetts Institute of Technology.

[16] Mayank Goel, Jacob Wobbrock, and Shwetak Patel. 2012. GripSense: Using Built-in Sensors to Detect Hand Posture and Pressure on Commodity Mobile Phones. 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, 545–554. https://doi.org/10.1145/2380116.2380184

[17] Emilio Granell and Luis A. Leiva. 2016. Less Is More: Efficient Back-of-Device Tap Input Detection Using Built-in Smartphone Sensors. 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, 5–11. https://doi.org/10.1145/2992154.2992166

[18] Jens Grubert, Matthias Heinisch, Aaron Quigley, and Dieter Schmalstieg. 2015. MultiFi: Multi Fidelity Interaction with Displays On and Around the Body. 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, 3933–3942. https://doi.org/10.1145/2702123.2702331

[19] Tiago Guerreiro, Ricardo Gamboa, and Joaquim Jorge. 2009. Mnemonical Body Shortcuts for Interacting with Mobile Devices. In *Gesture-Based Human-Computer Interaction and Simulation*, Miguel Sales Dias, Sylvie Gibet, Marcelo M. Wanderley, and Rafael Bastos (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 261–271.

[20] 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

[21] 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)*. Association for Computing Machinery, New York, NY, USA, 627–636. https://doi.org/10.1145/2047196.2047279

[22] Khalad Hasan, David Ahlström, Junhyeok Kim, and Pourang Irani. 2017. AirPanes: Two-Handed Around-Device Interaction for Pane Switching on Smartphones. 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, 679–691. https://doi.org/10.1145/3025453.3026029

[23] Juan David Hincapié-Ramos and Pourang Irani. 2013. CrashAlert: Enhancing Peripheral Alertness for Eyes-Busy Mobile Interaction While Walking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 3385–3388. https://doi.org/10.1145/2470654.2466463

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

[25] Ryosuke Hori, Ryo Hachiuma, Hideo Saito, Mariko Isogawa, and Dan Mikami. 2021. Silhouette-Based Synthetic Data Generation For 3D Human Pose Estimation With A Single Wrist-Mounted 360° Camera. In *2021 IEEE International Conference on Image Processing (ICIP)*. 1304–1308. https://doi.org/10.1109/ICIP42928.2021.9506043

[26] Hao-Juan Huang, I-Chao Shen, and Liwei Chan. 2020. Director-360: Introducing Camera Handling to 360 Cameras. In *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services* (Oldenburg, Germany) *(MobileHCI '20)*. Association for Computing Machinery, New York, NY, USA, Article 15, 11 pages. https://doi.org/10.1145/3379503.3403550

[27] Michael Xuelin Huang, Yang Li, Nazneen Nazneen, Alexander Chao, and Shumin Zhai. 2021. TapNet: The Design, Training, Implementation, and Applications of a Multi-Task Learning CNN for Off-Screen Mobile Input. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 282, 11 pages. https://doi.org/10.1145/3411764.3445626

[28] Dong-Hyun Hwang, Kohei Aso, Ye Yuan, Kris Kitani, and Hideki Koike. 2020. MonoEye: Multimodal Human Motion Capture System Using A Single Ultra-Wide Fisheye Camera. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 98–111. https://doi.org/10.1145/3379337.3415856

[29] S. Hwang, H. Jo, and J. Ryu. 2010. EXMAR: EXpanded view of mobile augmented reality. In *2010 IEEE International Symposium on Mixed and Augmented Reality*. 235–236. https://doi.org/10.1109/ISMAR.2010.5643584

[30] Masakazu Iwamura, Naoki Hirabayashi, Zheng Cheng, Kazunori Minatani, and Koichi Kise. 2020. VisPhoto: Photography for People with Visual Impairment as Post-Production of Omni-Directional Camera Image. 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, 1–9. https://doi.org/10.1145/3334480.3382983

[31] Tero Jokela, Jarno Ojala, and Kaisa Väänänen. 2019. How People Use 360-Degree Cameras. In *Proceedings of the 18th International Conference on Mobile and Ubiquitous Multimedia* (Pisa, Italy) *(MUM '19)*. Association for Computing Machinery, New York, NY, USA, Article 18, 10 pages. https://doi.org/10.1145/3365610.3365645

[32] Shunichi Kasahara, Shohei Nagai, and Jun Rekimoto. 2017. JackIn Head: Immersive Visual Telepresence System with Omnidirectional Wearable Camera. *IEEE Transactions on Visualization and Computer Graphics* 23, 3 (March 2017), 1222–1234. https://doi.org/10.1109/TVCG.2016.2642947

[33] Daehwa Kim, Keunwoo Park, and Geehyuk Lee. 2020. OddEyeCam: A Sensing Technique for Body-Centric Peephole Interaction Using WFoV RGB and NFoV Depth Cameras. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 85–97. https://doi.org/10.1145/3379337.3415889

[34] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[35] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell. 2010. A survey of mobile phone sensing. *IEEE Communications Magazine* 48, 9 (Sep. 2010), 140–150. https://doi.org/10.1109/MCOM.2010.5560598

[36] Gierad Laput and Chris Harrison. 2019. SurfaceSight: A New Spin on Touch, User, and Object Sensing for IoT Experiences. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300559

[37] 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

[38] Juyoung Lee, Hui-Shyong Yeo, Murtaza Dhuliawala, Jedidiah Akano, Junichi Shimizu, Thad Starner, Aaron Quigley, Woontack Woo, and Kai Kunze. 2017. Itchy Nose: Discreet Gesture Interaction Using EOG Sensors in Smart Eyewear. In *Proceedings of the 2017 ACM International Symposium on Wearable Computers* (Maui, Hawaii) *(ISWC '17)*. Association for Computing Machinery, New York, NY, USA, 94–97. https://doi.org/10.1145/3123021.3123060

[39] Frank Chun Yat Li, David Dearman, and Khai N. Truong. 2009. Virtual Shelves: Interactions with Orientation Aware Devices. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology* (Victoria, BC, Canada) *(UIST '09)*. Association for Computing Machinery, New York, NY, USA, 125–128. https://doi.org/10.1145/1622176.1622200

[40] Sugang Li, Xiaoran Fan, Yanyong Zhang, Wade Trappe, Janne Lindqvist, and Richard E. Howard. 2017. Auto++: Detecting Cars Using Embedded Microphones in Real-Time. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3, Article 70 (Sept. 2017), 20 pages. https://doi.org/10.1145/3130938

[41] Zhengqing Li, Shio Miyafuji, Erwin Wu, Hideaki Kuzuoka, Naomi Yamashita, and Hideki Koike. 2019. OmniGlobe: An Interactive I/O System For Symmetric 360-Degree Video Communication. In *Proceedings of the 2019 on Designing Interactive Systems Conference* (San Diego, CA, USA) *(DIS '19)*. Association for Computing Machinery, New York, NY, USA, 1427–1438. https://doi.org/10.1145/3322276.3322314

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

[43] Hyunchul Lim, Yaxuan Li, Matthew Dressa, Fang Hu, Jae Hoon Kim, Ruidong Zhang, and Cheng Zhang. 2022. BodyTrak: Inferring Full-Body Poses from Body Silhouettes Using a Miniature Camera on a Wristband. *Proc. ACM Interact.*

Mob. Wearable Ubiquitous Technol. 6, 3, Article 154 (sep 2022), 21 pages. https://doi.org/10.1145/3552312

[44] Hyunchul Lim, David Lin, Jessica Tweneboah, and Cheng Zhang. 2021. HandyTrak: Recognizing the Holding Hand on a Commodity Smartphone from Body Silhouette 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, 1210–1220. https://doi.org/10.1145/3472749.3474817

[45] Mona Hosseinkhani Loorak, Wei Zhou, Ha Trinh, Jian Zhao, and Wei Li. 2019. Hand-Over-Face Input Sensing for Interaction with Smartphones through the Built-in Camera. 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 32, 12 pages. https://doi.org/10.1145/3338286.3340143

[46] Zhihan Lv, Alaa Halawani, Shengzhong Feng, Haibo Li, and Shafiq Ur Réhman. 2014. Multimodal Hand and Foot Gesture Interaction for Handheld Devices. *ACM Trans. Multimedia Comput. Commun. Appl.* 11, 1s, Article 10 (Oct. 2014), 19 pages. https://doi.org/10.1145/2645860

[47] Fabrice Matulic, Riku Arakawa, Brian Vogel, and Daniel Vogel. 2020. PenSight: Enhanced Interaction with a Pen-Top Camera. 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/10.1145/3313831.3376147

[48] Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. 2017. VNect: Real-Time 3D Human Pose Estimation with a Single RGB Camera. *ACM Trans. Graph.* 36, 4, Article 44 (July 2017), 14 pages. https://doi.org/10.1145/3072959.3073596

[49] Florian Müller, Joshua McManus, Sebastian Günther, Martin Schmitz, Max Mühlhäuser, and Markus Funk. 2019. Mind the Tap: Assessing Foot-Taps for Interacting with Head-Mounted Displays. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300707

[50] Alejandro Newell, Kaiyu Yang, and Jia Deng. 2016. Stacked Hourglass Networks for Human Pose Estimation. In *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (Eds.). Springer International Publishing, Cham, 483–499.

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

[52] Henning Pohl and Michael Rohs. 2014. Around-Device Devices: My Coffee Mug is a Volume Dial. In *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices & Services* (Toronto, ON, Canada) *(MobileHCI '14)*. Association for Computing Machinery, New York, NY, USA, 81–90. https://doi.org/10.1145/2628363.2628401

[53] Philip Quinn, Seungyon Claire Lee, Melissa Barnhart, and Shumin Zhai. 2019. Active Edge: Designing Squeeze Gestures for the Google Pixel 2. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300504

[54] Helge Rhodin, Christian Richardt, Dan Casas, Eldar Insafutdinov, Mohammad Shafiei, Hans-Peter Seidel, Bernt Schiele, and Christian Theobalt. 2016. EgoCap: Egocentric Marker-Less Motion Capture with Two Fisheye Cameras. *ACM Trans. Graph.* 35, 6, Article 162 (Nov. 2016), 11 pages. https://doi.org/10.1145/2980179.2980235

[55] Yong Rui, Anoop Gupta, and J. J. Cadiz. 2001. Viewing Meeting Captured by an Omni-Directional Camera. 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, 450–457. https://doi.org/10.1145/365024.365311

[56] D. Scaramuzza, A. Martinelli, and R. Siegwart. 2006. A Toolbox for Easily Calibrating Omnidirectional Cameras. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 5695–5701. https://doi.org/10.1109/IROS.2006.282372

[57] Eldon Schoop, James Smith, and Bjoern Hartmann. 2018. HindSight: Enhancing Spatial Awareness by Sonifying Detected Objects in Real-Time 360-Degree Video. 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/10.1145/3173574.3173717

[58] Shaikh Shawon Arefin Shimon, Sarah Morrison-Smith, Noah John, Ghazal Fahimi, and Jaime Ruiz. 2015. Exploring User-Defined Back-Of-Device Gestures for Mobile Devices. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Copenhagen, Denmark) *(MobileHCI '15)*. Association for Computing Machinery, New York, NY, USA, 227–232. https://doi.org/10.1145/2785830.2785890

[59] Yu-Chuan Su and Kristen Grauman. 2017. Learning Spherical Convolution for Fast Features from 360° Imagery. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 529–539.

[60] Mingxing Tan and Quoc Le. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 6105–6114. https://proceedings.mlr.press/v97/tan19a.html

[61] Yu-Chih Tung and Kang G. Shin. 2016. Expansion of Human-Phone Interface By Sensing Structure-Borne Sound Propagation. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services* (Singapore, Singapore) *(MobiSys '16)*. Association for Computing Machinery, New York, NY, USA, 277–289. https://doi.org/10.1145/2906388.2906394

[62] Eduardo Velloso, Dominik Schmidt, Jason Alexander, Hans Gellersen, and Andreas Bulling. 2015. The Feet in Human–Computer Interaction: A Survey of Foot-Based Interaction. *ACM Comput. Surv.* 48, 2, Article 21 (Sept. 2015), 35 pages. https://doi.org/10.1145/2816455

[63] K. Wang and S. Lai. 2019. Object Detection in Curved Space for 360-Degree Camera. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 3642–3646. https://doi.org/10.1109/ICASSP.2019.8683093

[64] Tianyu Wang, Giuseppe Cardone, Antonio Corradi, Lorenzo Torresani, and Andrew T. Campbell. 2012. WalkSafe: A Pedestrian Safety App for Mobile Phone Users Who Walk and Talk While Crossing Roads. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications* (San Diego, California) *(HotMobile '12)*. Association for Computing Machinery, New York, NY, USA, Article 5, 6 pages. https://doi.org/10.1145/2162081.2162089

[65] Raphael Wimmer and Sebastian Boring. 2009. HandSense: Discriminating Different Ways of Grasping and Holding a Tangible User Interface. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction* (Cambridge, United Kingdom) *(TEI '09)*. Association for Computing Machinery, New York, NY, USA, 359–362. https://doi.org/10.1145/1517664.1517736

[66] Pui Chung Wong, Hongbo Fu, and Kening Zhu. 2016. Back-Mirror: Back-of-Device One-Handed Interaction on Smartphones. In *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications* (Macau) *(SA '16)*. Association for Computing Machinery, New York, NY, USA, Article 10, 5 pages. https://doi.org/10.1145/2999508.2999522

[67] Erwin Wu, Ye Yuan, Hui-Shyong Yeo, Aaron Quigley, Hideki Koike, and Kris M. Kitani. 2020. Back-Hand-Pose: 3D Hand Pose Estimation for a Wrist-Worn Camera via Dorsum Deformation Network. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 1147–1160. https://doi.org/10.1145/3379337.3415897

[68] Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, and William Freeman. 2012. Eulerian Video Magnification for Revealing Subtle Changes in the World. *ACM Trans. Graph.* 31, 4, Article 65 (jul 2012), 8 pages. https://doi.org/10.1145/2185520.2185561

[69] Bin Xiao, Haiping Wu, and Yichen Wei. 2018. Simple Baselines for Human Pose Estimation and Tracking. In *Computer Vision – ECCV 2018*, Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (Eds.). Springer International Publishing, Cham, 472–487.

[70] 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

[71] W. Xu, A. Chatterjee, M. Zollhöfer, H. Rhodin, P. Fua, H. Seidel, and C. Theobalt. 2019. Mo2Cap2: Real-time Mobile 3D Motion Capture with a Cap-mounted Fisheye Camera. *IEEE Transactions on Visualization and Computer Graphics* 25, 5 (May 2019), 2093–2101. https://doi.org/10.1109/TVCG.2019.2898650

[72] Wataru Yamada, Hiroyuki Manabe, and Daizo Ikeda. 2018. CamTrackPoint: Camera-Based Pointing Stick Using Transmitted Light through Finger. 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, 313–320. https://doi.org/10.1145/3242587.3242641

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

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

[75] Feng Zhang, Xiatian Zhu, Hanbin Dai, Mao Ye, and Ce Zhu. 2020. Distribution-Aware Coordinate Representation for Human Pose Estimation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 7091–7100. https://doi.org/10.1109/CVPR42600.2020.00712

[76] Xiang Zhang, Kaori Ikematsu, Kunihiro Kato, and Yuta Sugiura. 2022. ReflecTouch: Detecting Grasp Posture of Smartphone Using Corneal Reflection Images. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 289, 8 pages. https://doi.org/10.1145/3491102.3517440

[77] Yang Zhang, Michel Pahud, Christian Holz, Haijun Xia, Gierad Laput, Michael McGuffin, Xiao Tu, Andrew Mittereder, Fei Su, William Buxton, and Ken Hinckley. 2019. Sensing Posture-Aware Pen+Touch Interaction on Tablets. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3290605.3300285

[78] Jingjie Zheng and Daniel Vogel. 2016. Finger-Aware Shortcuts. 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, 4274–4285. https://doi.org/10.1145/2858036.2858355

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

A COMPARISON BETWEEN FISHEYE AND CUBEMAP PROJECTION FOR BODY POSE ESTIMATION

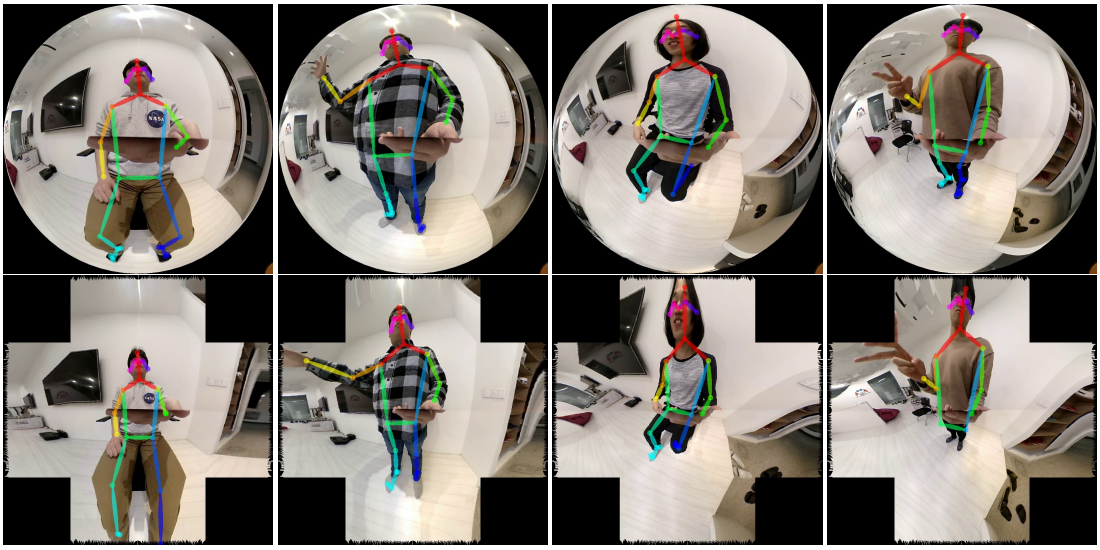
Our observations suggest that existing body pose estimation model works well with the screen-captured fisheye image without requiring further modification. However, we want to evaluate whether the undistortion step we applied has any significant advantage, because if there is none, then it is unnecessary to waste CPU resources on performing the undistortion.

In order to examine this, we followed the protocol in MeCap [1], where we collected 144 images consisting of two different postures, three gestures and two phone-holding orientations, from four participants. We asked the participants to vary their posture (standing and seated), keeping still and make some movements (hand and leg), and hold the phone in different orientations (tilted towards and away from the user). From these 144 captured fisheye images, we generate the undistorted images using cubemap projection, yielding another 144 images. In total, these 288 images were manually annotated with body keypoints and then compared to OpenPose's predicted joints keypoints (Figure 15) to derive error statistics. The error measurements are normalized by participant's shoulder width [1]. We did not compare with perspective and equirectangular format because the distortion around the edge is too high for any meaningful comparison.

A.1 Results and Discussion

The results show that, in overall, the measured error difference is minimal between the two image formats, both with more missing joints and higher normalized error for the lower body parts, as shown in Table 3. Especially for the foot (toe and heel), they are frequently not detected by both OpenPose and human annotators, as they appear too small and frequently occluded when seated, which is highly challenging even for a human annotator. For both types of image formats, as the participants held the phone using the non-dominant hand (left), the joints on this side have a higher misalignment error rate due to i) occlusion by the hand holding the phone and ii) the wrist joint holding the phone is too close to the camera. In overall, there is no stand-out approach. We can summarize three key insights.

- In some fisheye images, the body tracking is lost abruptly for the majority of the body parts. This happens in only 3 out of the 144 images (2%) we collected and we visually confirmed this by re-running OpenPose over the video file. This is mainly due to barrel distortion of the fisheye, whereas cubemap does not suffer from this.
- When the phone is tilted towards users, both formats suffer from the issues of i) shortened leg and ii) occlusion by hand, where OpenPose fails to estimate the lower body parts (Figure 15 (d)) and sometimes confused the gripping hand for leg. In this case, fisheye is slightly better.
- When users perform a kicking gesture, the leg extends to the edge of the fisheye image, which suffers from more distortion and hence causes lost tracking of the leg joints. In this case, cubemap projection is slightly better. Yet, cubemap suffers from its own issues, because it was cropped slightly for undistortion, hence the leg part appears bent and cut



(a) Seated | Flat (b) Standing | Flat (c) Seated | Tilted (d) Standing | Tilted

Figure 15: Image examples captured for body pose estimation comparison using OpenPose (Top: Fisheye, Bottom: Cubemap).

Table 3: Result of OpenPose vs. ground truth annotation on the two image formats (fisheye vs. cubemap projection). Face includes Nose, UpperNeck and HeadTop joints. Upper includes Shoulder, Elbow and Wrist joints. Lower includes Hip, Knee and Ankle joints. Foot includes BigToe, SmallToe and Heel joints.

	Fisheye		Cubemap	
	% Captured	Mean Normalized Misalignment Error	% Captured	Mean Normalized Misalignment Error
Overall	83.00	0.08	79.86	0.09
Face	98.15	0.07	99.07	0.07
Left Eye/Ear	98.61	0.03	99.31	0.04
Right Eye/Ear	93.06	0.03	93.40	0.06
Left Upper	100.00	0.11	100.00	0.15
Right Upper	97.69	0.06	99.07	0.08
Left Lower	82.64	0.18	75.00	0.16
Right Lower	88.19	0.09	82.64	0.12
Left Foot	39.58	0.04	34.72	0.04
Right Foot	57.64	0.05	46.53	0.05

off when it is being extended. This is especially obvious in seating posture where joints below the knees are cropped and hence incorrectly detected by OpenPose (Figure 15 (a)).

To conclude, cubemap projection is able to maintain the correct size of the body, but when the user extends the leg extremely, it appears to bend and cropped at the seam between each side of the cubemap. In contrast, fisheye has better FoV coverage, but is slightly distorted near the edge, and has an unbalanced body size (big shoulder, short legs). Overall, in many scenarios that do not require leg tracking, both fisheye and cubemap work equally well.

Therefore, we can save CPU resources by not performing undistortion. While undistortion helps a little, the main issue is the high and tilted viewpoint from the camera, as users tend to hold the phone around chest height during mobile interaction. From this camera viewpoint, the upper body appears larger and the lower body appears smaller (Figure 15 (d)). Hence, this is not really a distortion problem, but a viewpoint problem. MeCap [1] appears to suffer from the same problem, where the authors did not evaluate the lower body parts (until the knee joint only).

Another issue is the hand gripping the phone occludes a portion of the body, causing the upper body to appear to be separated from the lower body. This confuses the existing body pose estimation model. We suggest the ideal solution is to train a custom model that works with this viewpoint (short leg), and deals with distortion and hand occlusion directly, such as Mo2Cap2 [71], thereby saving processing cost.