Addressing the Input Gap on Mobile Devices with Implicit Hand Sensing

Nathan DeVrio

CMU-HCII-26-112
August 2026

Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Chris Harrison, HCII, CMU, Chair
Scott Hudson, HCII, CMU
Nikolas Martelaro, HCII, CMU
Hrvoje Benko, University of Washington

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Copyright © 2026 Nathan DeVrio

Keywords: Wearables, Sensing, Input Techniques, Context-Awareness, Pose, Mixed Reality, Privacy, Human-Computer Interaction

Abstract

Computing devices have historically maintained a balance of input and output capabilities for positive and productive user experience. However, new and upcoming mobile devices, particularly mixed reality (XR) headsets and smart glasses, have made trade-offs that prioritize mobility and wearability to enable always available content display at the cost of usable input. While advances in display and computation have increased the potential of these devices, limitations in input continue to constrain what users can actually do with them. This has led to an input gap that threatens to increase frustration, confusion, and inefficiency when users are not given enough input capability to interact with the interfaces presented to them.

Prior research has overly focused on improving explicit input techniques, which ignores the physical context wearable devices are inherently situated in. In contrast, in this thesis, I explore an alternative approach: implicit input which uses sensors worn on the body to detect behaviors that occur naturally around devices. The primary part of the body that I focus on is the hands, as hands are the dominant means by which people interact with the world. I present three approaches that demonstrate how implicit input can expand interaction by enabling devices to understand what a user is touching, how they are moving, and which objects they are interacting with. For each approach, through several proof-of-concept systems I show the effectiveness of novel sensing for enabling new forms of implicit input. Together, these systems illustrate how sensing everyday hand and body behaviors can meaningfully augment existing input. I argue that the rise of AI-driven mobile applications further exposes the shortcomings of current input frameworks, which provide limited user context and keep devices largely reactive. As a whole, my research agenda positions implicit sensing of hand interactions as a foundation for more proactive, context-aware devices that both better leverage advances in AI and help close the growing input gap in mobile computing.

Acknowledgements

Foremost, to Chris Harrison, who has been the driving force behind all of my work: I could not have asked for a better advisor. You are kind, caring, entertainingly opinionated, and above all a true savant of all things HCI. I mean this deeply — I hope to emulate even a fraction of the professor you are.

I would also like to thank my full committee: Scott Hudson, Nikolas Martelaro, and Hrvoje Benko. Scott, thank you for bringing your deep experience to bear, asking the challenging questions that helped me tighten my arguments into a stronger thesis. Nik, your expertise in design sharpened my thinking about what kinds of knowledge this thesis generates and how its principles can help others design similar systems. Finally, Benko, you brought an outsider's industry perspective that pushed me to think about my work in new, more practical ways.

More broadly, I am indebted to all my collaborators, who helped shape the work in this thesis and without whom it would not have been possible. Of special note are my industry collaborators, Wolf Kienzle, Robert Boldu, Eric Whitmire, Brian Tsang, and many others from whom I learned a great deal about applying research to real-world problems.

I also want to thank my FIGLAB mates: Vivian Shen, Daehwa Kim, and Vimal Mollyn. Beyond being wicked smart and always keeping me on my toes, you brought humor, levity, and fun to every day I came in to work.

For boundless support, laughter, and joy, I want to thank my many friends and colleagues in the HCII and wider CMU community: Pranav Khadpe, Tejus Gupta, Luke Guerdan, Samantha Speer, Rishi Veerapaneni, Bardienus Duisterhof, Cella Sum, Tucker Rae-Grant, Franchesca Spektor, Jordan Taylor, Rose Chang, Sanika Moharana, Lisa Egede, Adinawa Adjagbodjou, Venkatesh Sivaraman, Noor Hammad, Yi-Hao Peng, and Violet Han. The struggles any one of us faced were struggles we all shared, and I could not have finished this journey without your wonderful village of kindness and support.

Closer to home, I want to thank my dad, my brother Austin, and my sister Tori, who have been my guiding compass throughout this journey. To my extended family, thank you for your advice and companionship over the years. You gave me much-needed motivation more times than I can count.

Last but not least, there are two people I want to thank for shaping the course of my life these past few years. The first is my late mom. You are my shining north star, forever the inspiration for everything I do and the person I want to be. The second is my partner in life, Alicia (Dr. DeVrio). You make me a better person, take me places, and always encourage me to live life to the fullest. Thank you!

Contents

1 Introduction **1**
- 1.1 Overview . . . 1
- 1.2 The Input Gap on Mobile Devices . . . 2
- 1.3 Utilizing User Context . . . 3
- 1.4 Closing the Input Capability Gap . . . 4
  - 1.4.1 Conventional Approaches in Explicit Input . . . 4
  - 1.4.2 Improving Implicit Input . . . 4
- 1.5 Document Structure . . . 5

2 Background **6**
- 2.1 Surface Touch Tracking . . . 6
- 2.2 User Pose Estimation . . . 7
  - 2.2.1 Hand Pose Capture . . . 7
  - 2.2.2 Body Pose Capture . . . 8
- 2.3 Human Activity Recognition . . . 8
- 2.4 Held-Object Detection . . . 9

3 My Implicit Input Approaches **11**
- 3.1 Touch . . . 11
- 3.2 Pose . . . 12
- 3.3 Object Grasps . . . 12

4 Ad Hoc Touch Tracking Via Headset-Integrated Sensors **15**
- 4.1 Introduction . . . 15
- 4.2 Related Work . . . 16
  - 4.2.1 Detecting Touch with Instrumented Environments . . . 17
  - 4.2.2 Detecting Touch with Finger/Hand/Arm- Mounted Sensors 17
  - 4.2.3 Detecting Touch without Finger/Hand/Arm- Mounted Sensors . . . 18
  - 4.2.4 Detecting Touch using Shadows . . . 20
- 4.3 Implementation . . . 20
  - 4.3.1 Camera . . . 21
  - 4.3.2 Illumination Geometry . . . 21
  - 4.3.3 Driver Board . . . 21
  - 4.3.4 Video Stream . . . 22

4.3.5 Extraneous Shadow Suppression . . . 22
4.3.6 Finger Tracking . . . 22
4.3.7 Machine Learning . . . 24
4.3.8 Model Training Protocol . . . 24
4.3.9 Compute and Power Consumption . . . 24
4.3.10 Compatibility with Existing Headsets . . . 25
4.4 Data Collection Protocol . . . 25
4.5 Results and Discussion . . . 27
4.5.1 Across Illumination Geometries . . . 27
4.5.2 Across Surface Materials . . . 28
4.5.3 Across Lighting Conditions . . . 29
4.5.4 Across Surface Orientations . . . 29
4.5.5 On-Skin Touch Detection . . . 29
4.5.6 Supplemental Study: Hover Distance . . . 30
4.5.7 Supplemental Study: Multitouch . . . 31
4.6 Limitations . . . 31
4.7 Summary . . . 33

5 Arm Pose Estimation Using Commodity Smart Devices 34
5.1 Introduction . . . 34
5.2 Related Work . . . 35
5.2.1 UWB-Based Pose Tracking . . . 35
5.2.2 IMU-Based Wearable Pose Tracking . . . 36
5.2.3 Fusing Inertial and Distance Measurements for Wearable Pose Tracking . . . 37
5.3 Implementation . . . 37
5.3.1 Proof-of-Concept Devices . . . 38
5.3.2 UWB Ranging & IMU Data . . . 38
5.3.3 UWB Correction . . . 38
5.3.4 Pose Estimation . . . 39
5.3.5 Model Training . . . 40
5.3.6 Model Inference and SmartPoser Output . . . 41
5.4 Evaluation . . . 41
5.4.1 Daily Activities Procedure . . . 41
5.4.2 Gross Motor Arm Pose Procedure . . . 42
5.4.3 Data Capture & Ground Truth . . . 42
5.5 Results & Discussion . . . 43
5.5.1 Train/Test Procedure . . . 43
5.5.2 Pose Tracking Accuracy . . . 43
5.5.3 Spatial Error . . . 44
5.5.4 UWB Ablation Study . . . 44
5.5.5 Comparison to Prior Work . . . 45
5.6 Example Uses . . . 46
5.7 Limitations . . . 47
5.8 Summary . . . 48

6 Multi-View Depth Sensing From the Wrist **49**
6.1 Introduction . . . 49
6.2 Related Work . . . 51
6.2.1 Applications of Hand & Arm Pose Tracking . . . 51
6.2.2 Wrist-Borne Indirect Sensing . . . 51
6.2.3 Wrist-Borne Direct Sensing . . . 52
6.3 Implementation . . . 53
6.3.1 Hardware . . . 53
6.3.2 Firmware . . . 54
6.3.3 Compositing Multi-View Point Clouds . . . 55
6.4 Example Uses & Implementations . . . 55
6.4.1 Arm Pose Tracking . . . 56
6.4.2 Hand Pose Tracking . . . 57
6.5 Evaluation Procedure . . . 58
6.6 Results & Discussion . . . 59
6.6.1 Within-Session Performance . . . 59
6.6.2 Across-Session Performance . . . 60
6.6.3 Cross-User Performance . . . 61
6.7 Future Use Cases . . . 61
6.7.1 Bimanual Activities . . . 62
6.7.2 Ad Hoc Touch Tracking . . . 62
6.7.3 Held Objects . . . 63
6.7.4 Environments . . . 64
6.7.5 Body Scanning . . . 64
6.8 Limitations . . . 64
6.9 Summary . . . 66

7 Detecting Object Grasps With Sensor Fusion On the Wrist **67**
7.1 Introduction . . . 67
7.2 Related Work . . . 69
7.2.1 Object-Centered Activity Recognition . . . 69
7.2.2 Grasp Type Classification . . . 70
7.2.3 Object Grasp Detection . . . 70
7.2.4 Sensor Fusion for Planning Prosthetic Hand Grasps . . . 72
7.3 Implementation . . . 72
7.3.1 Sensor Selection . . . 73
7.3.2 Camera Sensor Details . . . 74
7.3.3 Power Consumption . . . 75
7.3.4 Comparison to Other Camera Sensors . . . 76
7.3.5 Camera Position Optimization . . . 77
7.3.6 Data Processing . . . 79
7.3.7 Model Architecture . . . 80
7.3.8 Ground Truth Data . . . 81
7.4 Evaluation Procedure . . . 82
7.5 Results . . . 83

7.5.1 Calculation of System Performance . . . 85
7.5.2 Cross-Session Performance . . . 85
7.5.3 Cross-Participant Performance . . . 86
7.5.4 Per-Object Performance . . . 87
7.5.5 Sensor Ablation Analysis . . . 87
7.6 Discussion . . . 89
7.6.1 Future Work for Grasp Applications . . . 89
7.6.2 Limitations . . . 90
7.6.3 Implications for Practitioners . . . 91
7.7 Summary . . . 92

8 Context Recognition Using Low-Power Sensor Triggers 93
8.1 Introduction . . . 93
8.2 Related Work . . . 95
8.2.1 Egocentric Activity Recognition . . . 95
8.2.2 Held-Object Recognition and Hand-Object Interaction Sensing . . . 96
8.2.3 Object Grasp Detection . . . 97
8.3 Implementation . . . 98
8.3.1 Sensor Selection . . . 99
8.3.2 Ring Hardware . . . 101
8.3.3 Smart Glasses Emulation Hardware . . . 102
8.3.4 Power Consumption Analysis . . . 102
8.3.5 Latency . . . 103
8.3.6 Ground Truth Data . . . 103
8.3.7 Data Processing . . . 104
8.3.8 Machine Learning Architecture . . . 104
8.3.9 Model Training Protocol . . . 106
8.3.10 Context Recognition Demo . . . 106
8.4 Evaluation . . . 106
8.5 Results & Discussion . . . 108
8.5.1 Calculation of System Performance . . . 108
8.5.2 Model Hyperparameter Tuning . . . 109
8.5.3 Comparing Cross-Session and Cross-User Results . . . 110
8.5.4 Sensor Ablation . . . 111
8.5.5 Performance-Power Tradeoff . . . 113
8.6 Applications . . . 114
8.6.1 Habit & Dietary Logging . . . 114
8.6.2 Medication Adherence . . . 114
8.6.3 Guided Task Completion . . . 114
8.6.4 Memory Support . . . 115
8.6.5 Dictation for Low-Vision Users . . . 116
8.6.6 Shopping Assistant . . . 116
8.7 Limitations & Future Work . . . 116
8.7.1 Additional Worn Device . . . 116

8.7.2 Grasps with Non-Worn Hand . . . 117
8.7.3 Non-Standard Grasps . . . 117
8.7.4 Complex Grasps & Environments . . . 117
8.7.5 Form Factor . . . 118
8.8 Summary . . . 118

9 Design Recommendations for Wearable Hand Sensing 119
9.1 When Sensor Fusion Helps . . . 119
9.2 Best Single Sensor: A Low-Resolution Camera . . . 120
9.3 Best Sensor Fusion: Multiple Cameras . . . 121
9.4 Promising Alternatives for Constrained Contexts . . . 121
9.5 Toward Proactive, Context-Aware Devices . . . 122

10 Conclusion 124
10.1 Thesis Contributions . . . 125
10.2 Future Directions . . . 126
10.2.1 General-Purpose Implicit Input . . . 126
10.2.2 Continuous Implicit Input . . . 126
10.2.3 Timely Implicit Input . . . 127

Bibliography 128

List of Figures

1.1 Design space of mobile devices showing how devices have increasingly prioritized mobility and slim form factors at the cost of usable input capabilities. Devices that retain mobility while improving input (shaded red area) are currently lacking, which has led to an **input gap**. 3

3.1 EclipseTouch is a headset-integrated approach for touch input on ad hoc surfaces. Illuminators create shadows (1/2/3/4), which are used to estimate touch contact and hover distance. 13
3.2 DiscoBand is a smartwatch strap (A) with 16 multi-zone depth sensors angled outward and toward the hands (B), capturing a point cloud (C) that enables arm and hand pose tracking. 13
3.3 Contextra is a wrist-worn system that works with an egocentric camera, such as on smart glasses, to determine a user's context (left) by fusing data from IR cameras, EMG electrodes, and an IMU to detect object grasps (middle), enabling interactions such as habit tracking (right). 14

4.1 High-level overview of EclipseTouch's experiment hardware and software pipeline. 18
4.2 Examples of ambient shadow suppression output across a variety of materials, lighting types (incandescent, fluorescent tubes, CFL, LED, sun), and illumination conditions (diffuse, floodlight, point source). Note that shadows visible in visible light (RGB Reference) may be invisible in infrared and vice-versa. Figure 4.8 provides a step-by-step example of shadow suppression. 21

4.3 Overview of my shadow suppression process. My ambient subframe (A) captures shadows cast by ambient light sources. When an illuminator is on (LED 3 in this example), I cast a new additional shadow into the scene (B). Note this shadow can be weaker than ambient shadows, as is the case in this example (see also Figure 4.2). I then subtract the ambient frame from the LED-illuminated subframe (B-A). Having now subtracted ambient light, the frame becomes darker, and represents light only cast by the headset LED. To compensate for variable scene brightness (e.g., varying surface albedo and hand distance) I perform a final normalization — norm(B-A) — which accentuates the shadow. Note only a single shadow remains, the one cast by the headset LED. 23
4.4 I tested three lighting conditions in my study: bright, typical, and dark. For reference only, I show the scene as it appears to a standard RGB camera along the top row. The bottom row shows the LED 3 illuminated subframe at the end of my pipeline (not shown are LED 1, 2 and 4 subframes, but the results are equivalent). In short, EclipseTouch is reasonably agnostic to ambient lighting condition as it provides its own illumination. Note also how shadows cast from ambient sources are not visible in the processed frame due to shadow suppression. 23
4.5 XR headsets on the market already include integrated infrared illuminators and infrared-sensitive cameras, suggesting EclipseTouch could be enabled via a software update. Additionally, I note that future AR glasses could also incorporate a single camera and LED in opposite corners of the frame, much like Ray-Ban Meta AI Glasses do today. 23
4.6 The twelve materials I tested in my study. Note the difference in appearance in RGB vs. infrared color spaces. 25
4.7 Results across illuminator combinations. A colored dot indicates that the corresponding illuminator is active in that channel. The best combination (LED 3 & LED 4, operating in separate channels) is marked with a star. 26
4.8 Touch classification accuracy vs. surface material. 28
4.9 Touch classification accuracy vs. lighting condition (left) and surface orientation (right). 28
4.10 Example 250 ms sequence of an index finger descending to tap a surface. Note how the shadow evolves over time (illuminated by LED 3), converging towards the finger, and essentially disappearing upon contact. My model uses this visual information to predict hover distance. 30
4.11 Left: Predicted vs. true hover distance. Overall, EclipseTouch is fairly linear and accurate, up to about 10 cm (mean error of 6.9 mm). Right: Mean absolute hover distance error vs. true distance — note the slight upward trend in error. 30

4.12 Results from my multitouch supplementary study, broken out by inputting finger. 31
4.13 Based on my evaluation results, I created a final prototype featuring a single infrared emitter and camera. 32

5.1 SmartPoser fuses UWB distance and IMU data from an off-the-shelf smartphone and smartwatch (A) to track a wearer's real-time arm pose (B & C). 34
5.2 SmartPoser system overview. My model has 46.4k parameters in total, with 3k parameters for the UWB Correction model and 43.4k parameters for the Pose Estimator model. 38
5.3 Example user motion sequence over one minute. Raw UWB distance is plotted in blue — note the significant errors, far exceeding possible arm lengths, when the watch loses line-of-sight to the phone. My corrected UWB stream is plotted in green. Ground truth distance provided by an external Azure Kinect is shown in orange. 39
5.4 Example set of predicted poses. Vertex error (vs. Kinect ground truth) is color coded. 40
5.5 The 20 terminal arm poses utilized in my user study. Note that participants were asked to transition between these poses and all intermediate data was captured for evaluation (i.e., not just the terminal poses). 40
5.6 Median per-joint positional error was 11.0 cm averaged across all study participants for the elbow and wrist joints. For the gross motions phase, median per-joint positional error was 11.2 cm, 9.2 cm for just the elbow, and 13.2 cm for the wrist. For the real-world activities phase, median per-joint position error was 11.1 cm, 8.4 cm for just the elbow, and 13.7 cm for the wrist. Error bars are standard deviation (cm). 43
5.7 Calculated metrics for all arm joints for the combined data set of gross motions and daily activities. All metrics are reported in cm. 43
5.8 Left: Plot of corrected UWB distance vs. Kinect-derived distance (280k datapoints). Right: UWB Correction RNN performance across study datasets. Mean Euclidean error reduces from 21.0 cm to 8.6 cm on average. Error bars are standard deviation. 44
5.9 Cumulative distribution functions (CDF) of the elbow and wrist joint errors. 44
5.10 Three rows of heatmaps visualizing maximum absolute Euclidean distance error for raw UWB distances (i.e., not corrected), corrected UWB distances, and predicted wrist joint position. For all measurements, error (in meters) is calculated with respect to Kinect-derived ground truth. Note the scale for the raw UWB distances (top row) is different from the bottom two rows. 45

5.11 Ablation results showing the effectiveness of using both UWB and IMU data for pose estimation. Results are calculated on data from both phases of the user study. When adding UWB data to the model trained on IMU alone, median error for both joints decreases from 14.2 cm to 11.0 cm. 45

6.1 DiscoBand is a smartwatch strap (A-C) featuring sixteen ultra-small, multi-zone depth sensors (closeup of two sensors in B) looking outwards and towards the hands (green and red frustums in D), which capture a fisheye point cloud (E). Many interactive uses are possible, including arm and hand pose tracking. 49
6.2 A high-level overview of wrist-worn, hand tracking methods. I include reference system(s) for each method, which I then evaluate across a range of desirable qualities. Orange tildes are used when the capability is system-dependent (and less innate to the method). *Capable of low-power operation using the next generation of sensors [270]. 51
6.3 Two early DiscoBand prototypes made from rigid PCBs (A & B). A physical mockup of how DiscoBand could appear in a commercial form factor (C & D). 54
6.4 A photo and PCB layout of my final DiscoBand prototype, which can also be seen in Figures 1, 5-8, 13-16. 54
6.5 DiscoBand's depth sensors capture different views of the hand, which can be composited into a unified 3D point cloud. In this example sequence, the user uncurls their fingers, which is apparent in the point cloud data. 55
6.6 Four example arm poses (top row) alongside real-time output of DiscoBand's arm pose pipeline (bottom row). 56
6.7 Four stills of example hand poses (top row) alongside renders of the 21 hand pose joints predicted by my model (bottom row). . . 56
6.8 Left: photo of example hand pose. Right: the three virtual view depth maps my pipeline creates as features for hand pose estimation. 56
6.9 The ten terminal arm poses used in my study. Top row: Reference pose images. Bottom row: point cloud of body as seen by the eight outward-facing sensors. 57
6.10 The ten terminal hand poses used in my study. Top row: Reference pose images. Bottom row: palm-normal-view synthetic depth maps **(see Figure 6.8)** of the hand as captured by the eight hand-facing sensors. 57
6.11 Hand pose study per-joint position error averaged across all participants for 21 hand keypoints. My lowest mean per-joint position error was 11.7 mm for within-session. 59

6.12 Arm pose study MPJPE across all participants for six upper body keypoints. Within-session accuracy was the strongest, with a MPJPE of 8.8 cm for the arm joints (keypoints 0 & 1) and 5.9 cm for all upper body points. 60
6.13 In these paired examples, I can see how DiscoBand is able to image the other (uninstrumented) arm. For illustration, the real point clouds have been rotated to match a greyscale reference photo. . 61
6.14 Example point clouds of various hand-held objects as captured by DiscoBand's eight hand-facing depth sensors. For illustration, the real point clouds have been rotated to match a greyscale reference photo. 62
6.15 Example single-frame point clouds as the user approaches a laptop to begin work. Green and red point clouds are from outward- and hand-facing depth sensors, respectively. In the right two examples, the hand-facing sensors capture the geometry of the keyboard and screen of the laptop. 62
6.16 As the user turns, a higher-resolution, multi-frame point cloud of the office is built up. Note the ajar door to the left is captured. . . 63

7.1 High-level overview of highly-related worn grasp detection systems. Comparable numbers drawn from published materials as best possible, but please consult individual papers for specifics. . . 69
7.2 Contextra system diagram. I take 30 × 30 optical data from three mini active IR cameras and fuse it with 16 EMG channels and IMU acceleration. These sensing streams are combined in my model to perform object grasp detection. 73
7.3 Timeline of a user grasping an object, along with demonstrative traces of sensor data from my different sources. In each frame, I highlight how different sensing channels (optical, EMG, and acceleration) can contribute by sensing different features of the grasp. For example, the motion of reaching for an object can be sensed by the accelerometer, then the object can be imaged by the cameras when opening the grip, and finally EMG can sense muscle contraction when lifting the object. 74
7.4 Example data from the three wrist cameras and mappings to their locations on the wrist; captured while picking up a paper cup from a table. These views show what kind of features can be picked up by each camera. 76
7.5 Views from the final camera positions of the thumb, palm, and pinky during a run of my simulator. On the right I show a rendering of the grasp pose looked like for each of the four test objects shown. 77

7.6 A diagram of my model architecture. My model was split into two branches based on ground truth label structure. For camera data, I used frame-based labels which indicated a touch or no touch state and for EMG and IMU data, I used event-based labels which indicated when grasp events (pick up or set down) occurred. EMG and IMU data were combined using a middle-fusion approach and then passed through a heuristic boost function to convert its output to match the frame-based labels. Once in a common format, the outputs from both branches were combined and returned with the frame-based labels. 79
7.7 The set of eight objects used in my user study. Objects were chosen in part to parallel prior work and in part to offer a diversity of sizes, weights, and reflectances. 82
7.8 Average precision and recall were 0.87 (SD=0.09) and 0.86 (SD=0.07) for cross-session and 0.82 (SD=0.12) and 0.84 (SD=0.08) for cross-participant, respectively. Recall tended to be better and lower variance. P2 and P5 had drops in precision in the cross-participant case, potentially from unique physiology that highlights the need for larger and more diverse training data. 84
7.9 Precision and recall were fairly consistent across objects, with precision varying only 0.03 from the mean for cross-session and recall only 0.04. Performance was loosely correlated with the size of objects because small ones were harder to image. Small objects like the USB and pencil performed worst while large ones like the bottles and notepad performed best. 84
7.10 Ablation analysis for the different sensing channels of Contextra. Results are shown for mean precision, recall, and IoU from the cross-session participant data. 86
7.11 A subset of the ablation analysis specifically on cross-user data to highlight the advantage of adding cameras to EMG. Results are shown for mean precision, recall, and IoU. 86
7.12 Examples of future applications for data collected from Contextra along with demonstrative example camera frames. Contextra could be used to detect micro gestures on the hand or objects (A), the identity of a particular object (B), or when a user's hand is near a surface for surface-based interactions (C). 89

8.1 TrigRing is a smart ring system that fuses information from a suite of onboard sensors to detect when a user interacts with an object. Designed to be paired with an egocentric camera device such as smart glasses, TrigRing (call-out) keeps the glasses asleep as the user reaches for an object (A), then uses its low-power, privacy-preserving sensors to detect the grasp and trigger the glasses (B), waking them only at the moments needed to provide contextual AI assistance (C). 94

8.2 Comparison of sensing modalities for ring-based grasp detection. Sensors are grouped into order-of-magnitude power bands. Duty cycling can move some sensors down by one to two bands. **Size** rates ring-scale feasibility: ● favorable, ◐ workable with engineering effort, ○ a first-order obstacle. **Complexity** rates the richness of the raw feature the sensor delivers: ● spatially or spectrally rich (an image, spectrogram, or multi-channel pattern), ◐ a short vector or waveform, ○ a single scalar or near-scalar reading. Sensors with greater than 10 mW of power were not considered for my TrigRing prototype (shown via vertical line). A **green check** ✓ indicates sensors I ultimately selected for testing. 99
8.3 Diagram of my prototype TrigRing smart ring system (see Figure 8.11 for my final version). I take image data from a mini active IR camera and fuse it with echo signatures from an ultrasonic acoustic sensor, motion signals from a 6-axis IMU, and light readings from a lux sensor. All of this data is processed and sent over serial via three parallel microcontrollers. Finally, a fisheye webcam was mounted to the side of the ring to capture reference images to help with debugging and sensemaking of my results. 100
8.4 Timeline of a user grasping an object, along with demonstrative traces of sensor data from my different sources. In each frame, I highlight how different sensing channels (mini IR camera, acoustic, IMU, and lux) can contribute by sensing different features of the grasp. For example, motion of reaching for an object can be sensed by the accelerometer, then the object can be imaged by the mini IR camera when opening the grip, and finally the acoustic echo pattern changes once the object is firmly held in the hand. . 101
8.5 A diagram of my ML model architecture. My model was split into four parallel branches, one for each sensing channel. Each branch used a different model for featurization of raw data before passing in the extracted features into a gated fusion block which learns intelligent combinations of the sensing channels. Finally, the output of the fusion block is passed through final linear layers to make an output grasp prediction. 105
8.6 Setup for my user evaluation. During the study, a set of 30 everyday objects (shown on right) were placed in sets of three on a sensing platform (shown on left) that used infrared detector pairs to collect ground truth data on when objects were picked up and set back down. Objects were chosen in part to parallel prior work and in part to offer a diversity of sizes, weights, and reflectances. 107

8.7 Plots detailing my pipeline hyperparameter tuning process. I tuned both the threshold for predicting a positive grasp class from the output of my model (left) and the number of seconds this positive class must be held to trigger a grasp event to the smart glasses (right). The final value for both hyperparameters (shown with the red circle) was selected as the value that maximized F1-score (i.e., maximally balancing precision and recall). Notable is that while practical performance of TrigRing is best when optimizing for recall, this significantly increases the null data false positive (FP) rate, thus I optimized F1-score instead. 109
8.8 Averaged precision, recall, and the null false positive (FP) rate across participant trials for the cross-session and cross-user cases. Precision was slightly higher and null FP rate lower (both good) in the cross-session case due to the inclusion of distinctive training data from that participant in the analysis. 110
8.9 Results for my ablation analysis of every possible sensor combination. Colored icons on the bottom indicate which sensor channels were used for that column. The best combination (all sensors) is marked with a star. 112
8.10 Comparison of the performance of different ablation sensor combinations and the total power consumption of the corresponding sensors for that combination. This comparison shows that the best combinations to maximize performance and minimize power use the mini IR camera, but not the acoustic sensor. 112
8.11 Photos of my final proof-of-concept ring after my power analysis. Compared to my earlier prototype, this ring has nearly identical performance, but consumes less than 25% of the power, runs wirelessly on a battery, and is significantly slimmer. 113
8.12 Example applications that TrigRing enables when paired with camera-equipped smart glasses. TrigRing tells the glasses when to capture an image and send it to an AI agent, which can then help with tasks such as logging that you took a medication (top left), tracking your diet (top right), guiding you through a DIY repair (bottom left), and remembering where you left your keys (bottom right). 115

List of Tables

4.1 Overview of key related work. Green is a positive attribute; red is negative. Refer to individual papers for study details. 17

7.1 Power consumption of each system component used for grasp detection. 75

8.1 Overview of key related work in wearable grasp detection. Performance metrics are derived from published materials where available. Values marked with an asterisk (*) represent my approximations based on reporting in the original text; please consult individual references for specifics. 95

8.2 Power consumption of each TrigRing component used in my first prototype. Power is dominated by the BLE stack on the MCU. With a lower data rate and sleep modes, MCU power can be as little as 1 mW. My final prototype (shown in Figure 8.11 and the Video Figure) consumes less than 25 mW. 102

Chapter 1

Introduction

1.1 Overview

Since the earliest computing systems, progress in human-computer interaction (HCI) has been shaped by a balance between what computers can present to users and what users can express to computers. Advances in output technologies, from text terminals and graphical displays to smartphones and immersive spatial headsets, have continually expanded the information, services, and experiences available through computing systems. Equally important, however, have been advances in input technologies. Keyboards first enabled textual communication, then mice enabled graphical interaction, and touchscreens transformed direct manipulation into a ubiquitous form of mobile computing. Throughout the history of computing, increases in output capability have typically been accompanied by corresponding advances in input, allowing users to effectively access and control increasingly powerful systems.

Today, computing is entering a new era of wearable and spatial devices. Mixed reality (XR) headsets, smart glasses, and AI pins promise access to digital information that is integrated with the physical world. These devices can present rich visual content, understand their surroundings, and deliver context-aware experiences that were once confined to science fiction. Yet despite remarkable advances in sensing, display, and computational capabilities, a fundamental challenge remains unresolved: input.

Unlike previous generations of computing platforms, emerging wearable devices often lack input techniques that are simultaneously efficient, socially acceptable, low effort, and available in everyday settings. Traditional approaches such as keyboards, mice, and touchscreens are difficult to translate to head-worn devices, while alternative approaches such as voice, handheld controllers, or mid-air gestures introduce their own practical limitations. As a result, the rapid evolution of output capabilities has outpaced corresponding advances in input, creating a growing interaction bottleneck for the next generation of computing.

1.2 The Input Gap on Mobile Devices

In Figure 1.1, I have imagined a design space of mobile devices comparing input capabilities to mobility. Observing this space shows that capabilities for input have shrunk while small form factors and mobility have significantly improved. This has led to the creation of an **input gap** where output is more capable than input. Devices with an unbalanced ratio of input and output offer a compromised user experience that cannot offer their true capabilities to their users. This is because when input or output is offered at a certain level in a system, users expect that they should be able to interact with or receive feedback from said system at a matching level. When expectations are misaligned, users may become frustrated, confused, or at best inefficient. As shown in Figure 1.1, designers have increasingly been prioritizing mobility and slim form factors at the cost of usable input capabilities.

For example, in the transition from the desktop, a non-mobile device, to the laptop, the first major mobile device, laptops sacrificed the mouse and keyboard functionality like the number pad. Similarly evolving from laptops to smartphones to smartwatches to smartglasses, devices lost physical keyboards, trackpads, and eventually touch screens entirely. While initial gains in improved mobility were high, there are now diminishing returns. Conversely, devices once maintained similar input to a desktop but are now significantly less capable, drifting down into the input gap. This is perhaps best typified by mixed reality headsets, such as Apple's Vision Pro, which has processing and output capabilities on-par with or even exceeding a desktop computing setup, but which is so input poor as to make writing an email or creating a slide show laborious. Future smart glasses with integrated displays will likely be at risk of a similar deficit where users can see more than they can do.

I believe this represents a fundamental shift in the design space of computing devices. In the past, devices were grounded in place and had ample considerations for both user input and content output. In contrast, on present and future trajectories, devices have chosen to be highly mobile and have slim form factors at the expense of removing common input affordances like keypads, touchscreens, and positional cursors.

For example, smartwatches are desirable because they are slim and allow users to see content like notifications without pulling out their phone. Yet when users want to perform meaningful interactions with said content, such as responding to a text, input on the watch is so constrained that they are forced to take out their phone which defeats the main benefit of the wearable.

I believe this "input gap" represents a grand challenge for the HCI community. Advances in mobile technology have given devices like smart glasses great potential, yet due to their compact form factor they lack traditional input like touchscreens or keyboards, which holds their potential back. If we could find better ways to interact with wearable devices, despite their small form factors, their full computational potentials could finally be unlocked. This is what my thesis aims to explore.

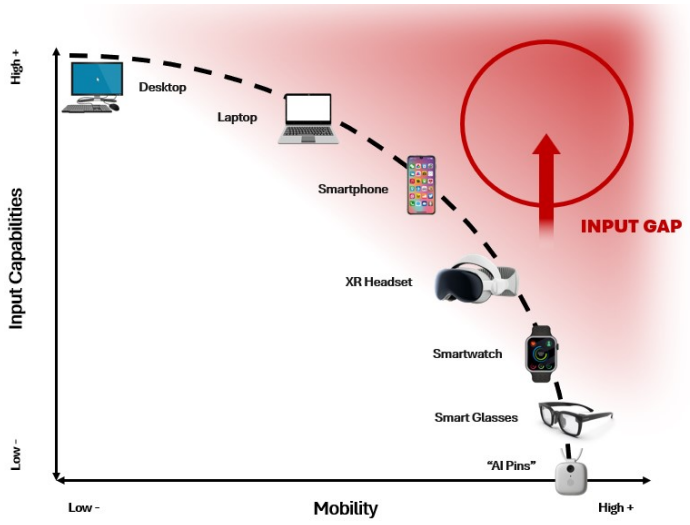


Figure 1.1: Design space of mobile devices showing how devices have increasingly prioritized mobility and slim form factors at the cost of usable input capabilities. Devices that retain mobility while improving input (shaded red area) are currently lacking, which has led to an **input gap**.

1.3 Utilizing User Context

To address the input gap, my work rests upon a critical observation: physical context is a powerful, yet underutilized source of information. Knowing what a person is doing, what they're holding, their mood, attention, and more are all valuable forms of input that are currently underutilized. New smaller devices may not be able to support traditional input, but if they have knowledge of a user's context, they could predict user intent and remove some of the input burden.

Conveniently, mobile devices are almost always worn, held, or carried on the body, and thus are in close proximity to many interesting interactions of body parts with the local environment. However, despite this powerful vantage point I believe very few devices take advantage of sensing the interactions directly surrounding them. This problem is especially stark when considering wearables such as a smartwatch, smart ring, or smart glasses ask users to put something additional on their body, but are not taking advantage of the vantage points available to them in their worn positions. For example, a smartwatch is worn on the forearm but tracks more information about the legs (e.g., step count) and lungs (e.g., respiratory rate) than it does about the arm and hand it is worn on. This is true for many other devices such as ear buds, smart rings, and smart glasses that users take and wear with them throughout many different activities. While a small and compact form factor is a limitation for traditional forms of input like touchscreens, keypads, or a mouse, sensors that can look outwards from devices into the surroundings have much greater latitude.

1.4 Closing the Input Capability Gap

My work uses **novel sensing techniques** to eliminate the tradeoff in the mobile device design space: allowing devices to keep mobile and slim form factors, but also taking advantage of sensing nearby body parts and context to enable powerful new forms of user input. Up until now, the missing link has been how to translate sensed body interactions into usable input for a device. Specifically, my work addresses how we can use hand interactions, as hands are the primary means we use our bodies to physically interact with the world.

1.4.1 Conventional Approaches in Explicit Input

One straightforward avenue is mapping hand interactions into explicit input commands. This includes all forms of deliberate input from the user intended to directly manipulate a UI or other virtual object or action. Examples include flicking your wrist to dismiss a message, tapping an icon, and pinching to zoom. This is a popular area of input research the HCI community has explored for close to the past half century, including myself in parts of this thesis, and for good reason.

Improving explicit input is important because it addresses the existing gaps with direct interaction on devices. For example, emerging categories of devices with head-mounted displays, such as mixed reality headsets and smart glasses, are refreshingly mobile and leave the hands unencumbered, but lack traditional input interfaces like a touch screen or buttons. This creates an input/output mismatch that makes these devices both difficult to use, and has contributed to poor user adoption despite revolutionary display capabilities. In order to become popular with consumers, it is thus necessary to innovate and provide more effective and efficient methods of explicit input.

1.4.2 Improving Implicit Input

However, a key issue with explicit input is that it doesn't supply any context about the user. This leaves input confined to the current paradigm of mobile systems that must act *reactively* to user commands. As we move into a future where our devices are acting increasingly as independent agents, another approach, which has up to this point been underexplored, is to improve input into the agents so that they can act more *proactively*. In contrast to traditional forms of conscious, explicit input, this requires input that is implicit and sensed automatically from the user without needing conscious intent.

Examples of implicit input include sensing natural human activities, such as taking a medication or prepping a kitchen, that provide useful signals about user tasks and behaviors. Implicit input gives us the ability to translate sensor data like video and audio into context like the pose of your arm or if you just picked up a tool. Yet, to maximize the potential of implicit input, systems need a way to translate that context into something actionable for an application to use. I believe that recent advances in AI are the perfect bridge for this.

Right now, AI has the potential to be a transformative technology for improving input, but its big problem is that it lacks contextual information about a user unless explicitly specified. This means current AI is not very useful for helping users with tasks out in the real world. To be useful, AI needs better contextual understanding of users. Using implicit input, we can generate basic descriptions of a user's context that can be used as input for an AI layer which uses its intelligence to tell the app what to do. My research has been exploring how to use implicit sensing to provide the best data for this channel.

In this thesis, by exploring implicit hand interactions, my work aims to simultaneously solve the practical shortcoming of wearable AI systems and use the combination of better context sensing and intelligence to solve the mobile input gap. This document presents my body of research toward this objective.

1.5 Document Structure

The next two chapters review prior work and outline the approaches I have explored for augmenting mobile input. Chapter 4 covers ad-hoc surface touch detection, describing my most promising technical approach: EclipseTouch. Chapter 5 describes my work on SmartPoser, a system that uses sensors on commodity smart devices to track the wearer's arm pose. Chapter 6 details my work on DiscoBand, a novel smartwatch sensing strap that used multi-view depth sensors to track hand pose, arm pose, and the nearby environment. In Chapter 7 I describe my work on Contextra, a wristband system for detecting object grasps with fusion of heterogeneous sensors. Chapter 8 describes TrigRing, a refined grasp detection system using sensor fusion on a smart ring, and introduces a unifying idea for using grasp signals as a trigger for continuous context recognition. To conclude, in Chapter 9 I synthesize findings from across my work to make design recommendations for future hand-sensing devices. Finally, in Chapter 10, I discuss the contributions of this thesis and propose several directions of future research.

Chapter 2

Background

My work on improving implicit input draws on techniques spanning several areas of human-computer interaction research, including touch tracking, pose estimation, human activity recognition, and context-aware computing. This chapter summarizes the prior work from these areas that most influenced my own.

2.1 Surface Touch Tracking

With the adoption of headsets that allow users to visualize virtual content overlaid on the physical world, users will naturally need a way to both interact with said content and have their devices understand interactions with the world. One neat solution to solving both of these problems is enabling surface touch tracking.

Right now the vast majority of mixed reality interactions constrain users to pointing at, poking, or tapping screens and buttons floating in midair. Without the built-in tactile feedback of a physical button or touch screen these interactions are distracting and less ergonomic. Grounding virtual user interfaces (UIs) to physical surfaces solves the tactile feedback problem, but introduces a new problem of detecting touch on surfaces.

In addition to interacting with mixed reality UIs, there are also other opportunities for using surface touch tracking to improve explicit input such as the ability to perform mouse or track pad input on any surface, including the body. This type of explicit input allows users to use positional tracking to control a cursor or perform gestures like swipes or pinches. One of the current drawbacks to using an XR headset as a general-purpose computing device is the lack of keyboard or cursor input. Surface touch tracking is an important opportunity for addressing this input disparity precisely because it provides the capability to enable both.

On the other hand, surface touch is also an unassuming source of implicit input. Users may make contact with a surface to interact with virtual content projected onto it, but they also may do so because they are interacting with a physical object or interface on that surface. If a system can recognize moments when surface touch occurs, this can be an implicit cue that some type of physical interaction or activity is occurring.

Multiple decades of HCI research have attempted to tackle the surface touch tracking problem. Importantly, while work on sensing touch from wearable devices has some overlap with research on detecting touch with instrumented surfaces, mobile touch detection is a unique and more challenging task. In multiple lines of work many different sensing techniques have been explored including egocentric cameras [86, 239, 273, 274, 318], IMUs [89, 90, 190, 209, 211, 260], acoustic [79, 100, 142, 185, 205, 259, 343], RF [139, 141, 292, 346, 347], and even indirectly from hand pose [63, 355]. Overall, the most promising modern techniques, such as EgoTouch [203] and UmeTrak [94], use egocentric cameras from a headset wearable and can reliably track touch with no prior calibration or instrumentation of the hands.

2.2 User Pose Estimation

2.2.1 Hand Pose Capture

Hands are perhaps the most natural and common way for users to interact with their devices and the world. Interactions with the hands are natural, intuitive, and introduce only a low cognitive load [128]. With a potential transition in the near-future to devices that are worn on the body and place no physical input affordances in the hand, it is vital to search for new ways to keep the hands engaged in input. The recognition of this problem is why hand pose capture has remained a central task in HCI research on input techniques for so long. It's also why I believe accurate hand pose capture represents the most foundational opportunity for improving implicit input.

My definition of hand pose capture broadly covers many different sub-problems. For example, it includes the more limited tasks of hand gesture recognition, pointing, or pinch detection which are all important forms of explicit input. However, it also includes the more challenging task of building a continuous 3D model of the hand that can be used for teleoperation of a robotic hand, or implicit sensing like detecting hand activities.

In the past, many different sensing techniques have been researched to capture hand pose from wearable devices. The technique used often differs depending on the application or vantage point the hands are being sensed from. Some popular sensing techniques have included electromyography (EMG) [137, 148, 173], tomography (EIT) [345], acoustic [95, 205, 264, 338], cameras [14, 46, 313], IMUs [246, 302, 320, 325, 326], capacitive [236, 284], and depth sensors [80, 81, 267]. Of these, camera sensing, in particular from an egocentric, head-based perspective has proven to be the most powerful technique with many high-performing prior works [93, 94, 114, 198], and even popular libraries such as MediaPipe [344]. However, hand pose capture faces a common sensing obstacle: the hand frequently occludes itself or is occluded by the arm. Camera-based systems are especially vulnerable to this, and their added power and privacy costs have kept research into alternative sensing techniques active.

2.2.2 Body Pose Capture

Zooming out beyond the hands, capture of the entire body also is a valuable opportunity for augmenting input. In certain contexts, such as posing a virtual avatar, having an accurate estimate of the user's body pose is a highly useful form of raw explicit input. Yet more abstractly, once a model of body pose is obtained, this information can then be used as a proxy about different behaviors, poses, or activities that may be of interest to a system. This second part underscores the full opportunity body pose captures can provide in augmenting implicit input on mobile devices.

Of these potential input opportunities for body pose capture, numerous health applications are notable. For example, repetitive leg motions can be used to count squat exercises, and changing arm reach can be used to track rehabilitation progress. While many of these have been attempted to be tracked by inertial motion sensors (IMUs) in the past [201, 254], such as the ones built into a smartwatch [201], IMUs have the limitation of only accurately tracking the point of instrumentation (e.g., the wrist for the smartwatch example). Any health tracking beyond the instrumented points is only an estimate, which is the main reason why full body pose capture has the potential to unlock many input benefits.

Counterintuitively, one popular approach for estimating body pose from wearables has been to use IMUs integrated into various devices worn across the body [67, 168, 240, 254, 290, 332]. The idea behind these works, such as IMUPoser [201], is that despite only sensing the motion of isolated body parts, physical models built upon deep learning can be used to fill in the rest. Outside of IMUs, other methods used in the past for estimating body pose include cameras [109, 118, 130], radio frequency (RF) waves including Wi-Fi [237, 238] and ultrawide-band (UWB) [191, 265, 328], acoustic [181], and specialized garments [339, 340].

2.3 Human Activity Recognition

Recognizing a user's local activity context offers a valuable form of implicit input, signaling both what the user is doing and how the system should respond. This context detection is a unique opportunity in its own right because it not only serves as implicit input directly, but also makes other forms of input far more useful. For example, in a life logging application, arm pose capture alone may recognize that a user is holding their hand up to their mouth. However, once augmented with context information, a device could go one step further to detect more specific actions including eating, brushing teeth, coughing, or just scratching an itch that each warrant different responses.

Many prior researchers have explored tracking human activities and behaviors through instrumented environments [123, 229, 230, 248, 299, 314], but when users leave these environments the tracking is lost. The defining characteristic of mobile devices is that they are mobile, and thus inherently users take them with them as they move around and perform many activities. For this reason, I believe

mobile devices in fact have an ideal perspective to sense and recognize human behaviors and contexts no matter where they are.

A significant amount of prior HCI and ubiquitous computing research has explored the problem of HAR and how wearable sensors can be used to recognize discrete classes of activities such as cleaning, cooking, or even specific steps of making a recipe. Most commonly, the sensing modalities used for wearable HAR are microphones [149,150,182,329] and IMUs [20,92,96,216,300], as audio and motion data contain the most accessible, salient information about a user's activity on mobile devices. Other less common sensing modalities have included on-body cameras [48, 163, 213, 227], RF [72, 298], and a fusion of multiple sensing modalities [1, 31, 158, 161, 196, 200, 232, 276, 331]. Of these, camera and fusion approaches appear to be the most promising for future research with the introduction of powerful vision-language models (VLMs) [220].

Yet, a persistent problem within HAR research has been knowing where and when to look for possible activities in a dense and continuous stream of sensor data. On a mobile device it is often too power and computationally expensive to constantly be performing such inference, especially on video data. Thus another important line of work searches for ways to use lighter-weight sensing to detect smaller contexts or "trigger moments" that may indicate an activity of interest [180, 200, 220, 331]. I have found this last problem especially compelling in my own work, since solving it is what I believe will unlock the benefits of HAR-based implicit input sensing for real-world users.

2.4 Held-Object Detection

A final goal for context-aware wearable computing is detecting the objects a user is holding. Objects reveal a great deal about the activity being performed, thus both detecting when a user is holding something and identifying what that object is have become prime directions of HCI research. This work falls into two major categories: held-object identification and object grasp detection. Systems in both categories have drawn on a range of sensing techniques, including EMG, MMG, capacitive sensors, optical sensors, and cameras.

Held-object identification is concerned with the problem of recognizing objects currently held in the hand. ViBand [153] for example used an oversampled accelerometer to recognize motion and vibrations typical of use with certain objects. EM-Sense [154] and Maekawa et al. [176] meanwhile used the electromagnetic radiation emitted by electrical objects to discern their identity. More uniquely, other work [37, 167] has used hand pose as an indirect "sensor". By using pose tracking to detect grasp shapes that are signature to use of certain objects, the identity of those objects can be determined.

Object grasp detection, by contrast, focuses on detecting the specific moments when a user interacts with an object. I believe this direction has strong potential for implicitly detecting changes in user context, and I have pursued several projects building on the wrist-wearable grasp detection work of Berlin et al. [29], Cofer et

al. [50], and Theiss et al. [282]. Of particular interest is these projects' shared use of a tiered sensing approach, in which multiple heterogeneous sensors work together, each successively activating another sensor with higher fidelity [29, 50, 282]. Sensor fusion is an extremely powerful approach for grasp detection, in addition to many other forms of implicit input, as I will discuss in Chapters 7 and 8.

Chapter 3

My Implicit Input Approaches

In this chapter, I summarize three key approaches I explored to improve implicit input on wearable devices: touch, pose, and object grasps. All three are motivated by the same underlying question: *what does a device need to know in order to proactively assist users based on their context?* My first approach holds that devices should know what a user is touching. Touch has long been explored as a form of explicit input, but knowing what or when a user touches something is also a valuable implicit signal. My second approach holds that devices should know how a user's body is moving, since pose and specific motions are highly characteristic of certain activities. My third approach holds that devices should know what object a user is holding, since picking up an object is itself a strong implicit signal that an interaction is about to occur. For each approach, I explored multiple techniques using novel wearable sensors to detect the behavior of interest.

3.1 Touch

One of the most fundamental ways to interact with the world around us is touch. Touch is not only how we manipulate objects, but making contact with an object is also a strong signal of intent to interact with that object. For this reason, touch is a valuable approach for sensing implicit hand interactions. Sensing touch can help devices answer questions like: are your fingers contacting a surface? How many and which ones? How much pressure are they applying?

Touch tracking has been explored in depth by the HCI community for decades, but mainly for explicit input purposes [97, 159, 273, 308, 318, 349]. Using touch as implicit input is a different approach that has been far less explored. In my work, I have explored tracking touch input with two different technical approaches. Both of these approaches were envisioned as addressing the longstanding problem of ad hoc surface touch detection on mixed reality (XR) headsets, which in the past have struggled with integrating virtual content with the physical world. I started with developing an approach VelociTrack [60], wherein I used a pair of high-speed cameras track the fingers and derive virtual motion signals from fingertip depth that can be used to detect decelerations upon touch. This technique worked well,

but struggles on softer surfaces that disrupt fingertip motion. In my second and complementary approach, EclipseTouch [202] (Figure 3.1), I solved this by using a different technique where I used infrared light illuminators to cast structured shadows of the fingertips that provide distinctive visual features to a camera of when touch contact occurs. Chapter 4 will discuss EclipseTouch in detail.

3.2 Pose

Another powerful signal for implicitly sensing interactions is the pose of a user. The way we move our bodies tells a lot about what we are doing. When people perform most activities or behaviors, their hand or arm adopts a pose that is characteristic of that activity. For example, pinched fingers when strumming a chord, a precision grip when holding a pencil, alternating straight and bent arms when doing dips, or a pulling motion when sawing wood. For all these examples, sensing pose can provide a strong implicit signal that an activity is being performed. As a result, devices should know things like: is your hand making a distinctive pose? Are you moving your arms? Are you sitting down or on the move?

The first technique I explored focused on tracking a user's full-body pose for games and social interactions in virtual reality (VR). This system, ControllerPose [4], placed two wide-angle cameras in the left and right VR controllers and pointed the cameras back at the body to image the torso, arms, and legs. In my next technique, I pivoted to investigating how smart devices users already use and wear can implicitly track the wearer's arm pose. In SmartPoser [59], I accomplished this by using commodity ultra-wideband (distance) and IMU (motion) sensors already built into a paired smartphone and smartwatch to estimate the pose of the watch arm. Chapter 5 will discuss SmartPoser in further detail. Finally, in DiscoBand [58] (Figure 3.2), I took a different approach and showed how the development of a smartwatch band with a novel multi-view depth sensing setup could be used to not only track arm pose, but also hand pose, and implicit interactions with the nearby environment. Chapter 6 will discuss DiscoBand in detail.

3.3 Object Grasps

Objects hold a great deal of contextual information and picking one up is a strong indicator of an intent to interact with it. Knowing what object you're manipulating, whether it be taking a medication or repairing a car, is an opportunity to be proactive and assist a user in that activity. I have discovered that grasping an object is a powerful implicit proxy that an interaction is occurring. Many common activities center around grasping objects. From drinking a cup of coffee, to going for a drive, or playing ultimate Frisbee, any automated system that wants to recognize these contexts should focus on using object grasps as a trigger that something interesting has happened. In addition, grasps tend to be a reliable proxy because users don't usually pick up objects unless they intend to interact with them.

Object grasps are complicated and thus benefit from multiple types of sensors working together to detect them. I have pursued two consecutive approaches to this problem. The first, Contextra [57] (Figure 3.3), is a wristband system that detects grasps by fusing IMU data, electromyography (EMG) data, and data from three mini infrared cameras. Chapter 7 discusses Contextra in detail. The second, TrigRing, builds on the ideas from Contextra to create a constellation system: a wearable that detects object grasps triggers a wake-up signal to smart glasses, which then use a camera and vision-language models (VLMs) to determine what object the user is holding or what activity they are engaging in. Running an egocentric camera continuously, as smart glasses do, consumes substantial power and can compromise user privacy. By offloading the grasp detection trigger to a separate, lower-power device, the glasses can instead remain asleep most of the time. TrigRing is a prototype implementation of this triggering concept, built around a low-power sensing smart ring; its details are discussed in Chapter 8.

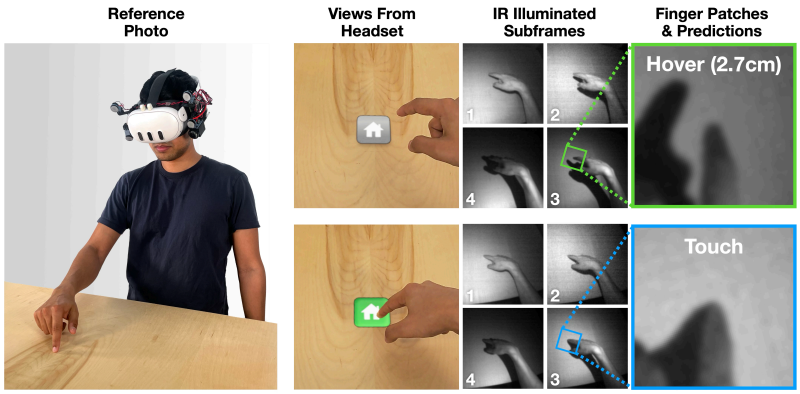


Figure 3.1: EclipseTouch is a headset-integrated approach for touch input on ad hoc surfaces. Illuminators create shadows (1/2/3/4), which are used to estimate touch contact and hover distance.

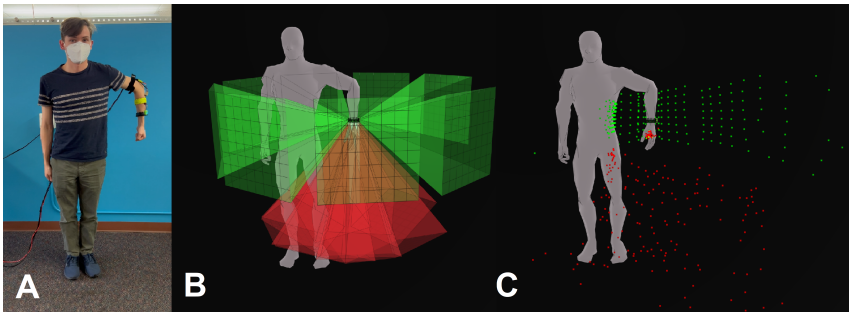


Figure 3.2: DiscoBand is a smartwatch strap (A) with 16 multi-zone depth sensors angled outward and toward the hands (B), capturing a point cloud (C) that enables arm and hand pose tracking.

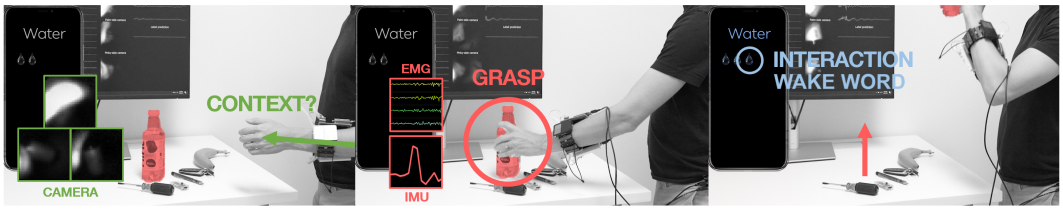


Figure 3.3: Contextra is a wrist-worn system that works with an egocentric camera, such as on smart glasses, to determine a user's context (left) by fusing data from IR cameras, EMG electrodes, and an IMU to detect object grasps (middle), enabling interactions such as habit tracking (right).

Chapter 4

Ad Hoc Touch Tracking Via Headset-Integrated Sensors

This chapter was adapted from my published paper:

> Vimal Mollyn*, Nathan DeVrio*, Chris Harrison. 2025. EclipseTouch: Touch Segmentation on Ad Hoc Surfaces using Worn Infrared Shadow Casting. *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '25).*

4.1 Introduction

Mixed reality (XR/AR) headsets are becoming more widespread, with increasing consumer excitement about forthcoming glasses-like form factors. However, navigating user interfaces on these devices generally requires users either to carry accessory controllers everywhere they go, or be limited to poking and swiping at interfaces in the air.

A complementary, long-envisioned interaction option is to ground virtual interfaces to real-world surfaces. One of the most reliable ways to achieve such interactions is to instrument surfaces with sensors. While robust, it is not feasible to do this for every surface in the world. In a future with highly mobile wearers of AR glasses, users may wish to temporarily and opportunistically appropriate surfaces for touch interaction. For this reason, there exists a significant body of work on ad hoc touch input without instrumentation of the environment. In most cases, this requires instrumenting the user instead, often with a special-purpose accessory device. Even when such sensors could be plausibly integrated into future smartwatches, it is most common for people to wear watches on their non-dominant hand.

I believe the ideal sensing method should integrate directly into the headset/glasses so that the user only needs to carry a single, self-contained device (i.e., the glasses). Only a handful of methods, which I discuss in Related Work, achieve this property. To this, I add two additional practical constraints: robust operation 1) across a wide variety of materials, including the user's skin; and 2) across environmental conditions (dark, bright, noisy, moving, etc.).

In this work, I present EclipseTouch, a new headset-integrated technique for ad hoc touch sensing. My system leverages the well-known phenomena of shadow casting, utilized in prior touch sensing work, but never demonstrated in a single worn device. More specifically, EclipseTouch uses an infrared, egocentric headset camera, which captures shadows cast by one or more synchronized infrared illuminators on the headset (Figure 4.1). As the geometry between the camera and illuminators are fixed, these shadows inherently capture a finger's distance from a surface, including direct contact (Figure 4.10). Importantly, my approach must first filter out shadows cast by extraneous light sources in order to be robust. At the core of my system is an optimized deep neural network that has an inference time of 0.47 ms on an Apple M2 processor (used in the Apple Vision Pro). The result is a method that works "out of the box" and requires no pre-registration or calibration of the environment, surface, or user. My approach works across a wide array of common surfaces, as well as lighting conditions, from bright to pitch-black. EclipseTouch can also be readily integrated into several popular XR headsets that already contain the requisite sensing hardware and compute. Taken together, this set of capabilities sets it apart from prior work, even those relying on similar phenomena.

4.2 Related Work

In this section, I review prior systems that have used techniques relevant to EclipseTouch. I begin with systems that examined the problem of ad hoc surface touch detection. For these, I start with instrumented environments (the systems least similar to my approach), progress to arm-worn mobile systems, and finally to mobile systems that do not require arm instrumentation (most similar to my approach). I conclude this section with a review of systems that specifically used shadows for touch tracking.

I emphasize that although active-illumination shadow tracking has been used previously — including for the exact same use case as this present work — my particular instantiation offers a favorable mix of capabilities not demonstrated in any prior work (Table 4.1), including:

- Uses hardware already present in modern headsets (cameras and illuminators).
- Requires no instrumentation of the user's arms (i.e., bare hands).
- Enables ad hoc input for commonplace surfaces, including the user's skin.
- Works across lighting conditions, including in complete darkness.
- Works "out of the box", requiring no pre-registration or calibration of the surface, user, or environment.
- Offers high touch input accuracy (98.0% touch segmentation, 6.9 mm hover distance estimation mean error).

System Name	Sensor Hardware	Mobile vs. Stationary	Instrumented Hands / Arms	Supports Multitouch	World Input	Skin Input	Demonstrated in Darkness	Estimates Hover Distance	Touch Accuracy
TapLight [273]	IR Camera, Structured Light	Mobile	No	No	Yes	No	No	No	95.3%
OmniTouch [97]	Depth Camera	Mobile	No	Yes	Yes	Yes	No	No	96.5%
EgoPressure [349]	RGB Camera	Mobile	No	Yes	Yes	No	No	No	n.r.
MRTouch [318]	Depth & IR Camera	Mobile	No	No	Yes	No	No	No	96.5%
PressureVision++ [85]	RGB Camera	Stationary	No	Yes	Yes	No	No	No	89.3%
PlayAnywhere [308]	IR Camera, IR LED	Stationary	No	No	Yes	No	No	No	n.r.
Matsubara et al. [186]	IR Camera, IR LEDs	Stationary	No	No	Yes	No	No	No	96.1%
EgoTouch [203]	RGB Camera	Mobile	No	Yes	No	Yes	No	No	94.9%
Shadow Touch [159]	RGB Camera, White LED	Mobile	Yes	Yes	Yes	No	No	No	99.1%
EclipseTouch (Ours)	**IR Camera, IR LEDs**	**Mobile**	**No**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes (6.9 mm error)**	**98.0%**

Table 4.1: Overview of key related work. Green is a positive attribute; red is negative. Refer to individual papers for study details.

4.2.1 Detecting Touch with Instrumented Environments

The most straightforward way to add touch tracking to a surface is to instrument that surface directly. This is a well-explored area of research that I will not cover in depth for brevity. Some of the most common techniques have included placing microphones [99, 215, 222] or LIDAR [152, 275] on the surface, or cameras above the surface. For cameras, there have been many different optical approaches. For example, thermal cameras have been used to detect changes in heat left behind on the surface after touch [77, 127, 147, 155]. RGB cameras have been used to capture images of the fingernail and its color change during presses [3, 47, 184, 277] or even estimate fingertip pressure [47, 85, 86]. Perhaps the most popular technique has been to use fixed depth cameras operating above surfaces. After Benko and Wilson's works that pioneered this approach [28, 309], many other iterations have improved performance using flood-fill algorithms [317], combining infrared camera data [316], and applying machine learning [71]. Other lesser used optical approaches include multi-path interference from infrared depth cameras [256,315], laser speckle imaging [221], and imaging reflections of the finger in mirrors and glossy surfaces [218,336]. A major drawback of all these systems is that they need to instrument every potential surface or environment that the user may wish to interact with, which scales poorly. For this reason, research has looked into instrumenting the user with sensors to detect touches on ad-hoc surfaces, which I review next.

4.2.2 Detecting Touch with Finger/Hand/Arm- Mounted Sensors

When a user touches a surface, their touching finger produces a number of characteristic signals that could be used to detect touch contact. For instance, IMUs attached to the fingernail can detect spikes in deceleration that occur when a finger touches a surface [210, 211, 260], but are cumbersome for users to wear and recharge. More popular is an IMU ring form factor, which could be used to detect touch [89, 90] and mouse-like 2D inputs [138, 160, 257]. Beyond rings, researchers have also explored using IMUs placed on the wrist (like a smartwatch) to detect tap events [190], however this signal is less robust.

Fingers also passively produce characteristic acoustic signals when tapping or swiping on a surface, which travel through the body and can be sensed with sensors mounted on the arm [79,100,142,185]. Alternatively, touches can be sensed

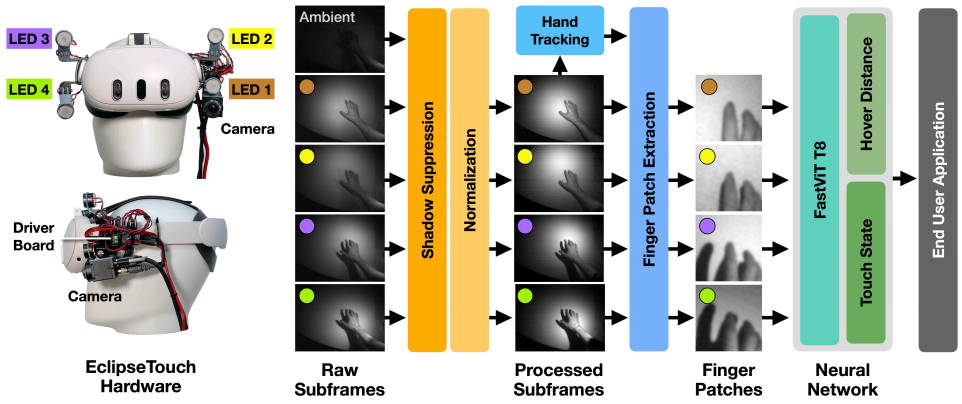


Figure 4.1: High-level overview of EclipseTouch's experiment hardware and software pipeline.

through changes in reflections of actively emitted acoustic signals. For instance, Mujibiya et al. [205], SoundTrak [343], and VersaTouch [259] emitted ultrasonic waves through transducers placed on the arm and finger to detect contact and pressure on the skin.

For detecting touches to the skin, one accurate approach that researchers have explored is detecting changes to RF signals transmitted through the user's body. AtaTouch [141] detected subtle finger pinches by sensing changes in impedance between an antenna and the user's body. SkinTrack [347] used two devices, a signal-emitting ring and a wristband receiver, with the body as an electrical waveguide to sense finger touch on the skin. ActiTouch [346] and ElectroRing [139] work in the same way, but move the transmitter and receiver to more convenient form factors. Finally, Z-Ring [292] used a single electrode ring that acted as a transceiver to sense bio-impedance changes in the hand. While these approaches are accurate, they are inherently limited to operation on conductive objects, such as the human body (but not most walls or furniture). A key limitation of all these prior systems is that they require instrumentation of the user's finger, hand or arm, in addition to the XR headset.

4.2.3 Detecting Touch without Finger/Hand/Arm- Mounted Sensors

For users that are already wearing an XR headset or glasses, rather than instrument a user's hands with a second device (likely special-purpose, as even smartwatches are rarely worn on the dominant hand), a more practical approach would be to have all necessary hardware contained within the XR headset/glasses. Dominant among these approaches is to use headset mounted cameras with computer vision to detect touches on surfaces.

As most modern headsets come with built-in 3D hand and world mesh tracking support, research has looked into inferring surface touch using this data. However, the world mesh modern headsets build is not currently accurate enough for touch detection purposes. For this reason, TriPad [63] required users to calibrate and define surfaces; hand tracking and dwells were used to instantiate touch planes and touches were detected by measuring fingertip proximity to the created plane.

Richardson et al. [239] and Streli et al. [274] focused on surface typing and used neural networks to analyze patterns of hand motion to detect surface taps with high accuracy. However, both these systems required pre-registered surfaces to work, and could not detect stateful touches (including hover and touch-ups).

Another approach that has seen the most success in the past is using depth cameras. OmniTouch [97] and Imaginary Phone [91] were early among these efforts and used depth cameras to track the fingers and detect touches on the palm and other surfaces using a combination of flood-filling and contour detection. MRTouch [318] combined depth sensor data with infrared reflectivity data from a Microsoft HoloLens to improve touch detection accuracy. The problem with depth cameras, even today, is noisy signal — the difference between a finger touching vs. slightly hovering above a surface is hard to distinguish. For this reason, OmniTouch required users to lift fingers 20 mm above the surface to reliably separate hovering from touch events. This is awkward, and not like how one scrolls or types on their touchscreen devices. Moreover, depth cameras have other drawbacks including higher power consumption, lower resolution and lower framerates. Taplight [273] estimated fingertip depth and surface contact by using remote vibrometry (laser speckle sensing) obtained from a headset mounted laser and monochrome camera. This approach, while accurate in some contexts, does not function on many common surface materials, and can additionally fail from user head motion and multiple inputting fingers.

Most relevant to EclipseTouch is recent work on detecting surface contact using only headset cameras. This is challenging, as the finger directly occludes the point of touch contact, and the system must adapt to surface materials, lighting conditions, touch types, and skin tones. PressureVision [86] and later PressureVision++ [85] described deep learning-based approaches to detecting contact pressure of the hand to a surface, albeit with a fixed desk-mounted camera and restricted lighting and surface conditions. EgoPressure [349] extended this work to headset-mounted cameras by collecting a large egocentric hand pressure estimation dataset, but their system ran offline (not in real-time) and required ambient illumination. In my prior work EgoTouch [203], I demonstrated a system for detecting on-skin touch and force using only headset RGB cameras, by looking at patterns of skin deformation, color change and shadow convergence. Similar to EgoTouch, PalmPad [102] could also detect touches to the skin with a headset-mounted RGB camera, however it could not estimate force and only supported the palm. Both EgoTouch and PalmPad ran in real-time, but only worked on the skin, required ambient illumination (i.e. did not work in the dark), and were susceptible to false positives from extraneous shadows in the environment.

The fundamental problem with all these prior approaches is that they rely on existing sources of illumination. However, ambient light is uncontrolled — it can be diffused, harsh, oblique, bright, multicolored, single/multi-point, and even non-existent (dark). With EclipseTouch, I sought to overcome these limitations by controlling the shadows cast by the finger on the surface.

4.2.4 Detecting Touch using Shadows

To conclude my literature review, I now specifically discuss systems that have leveraged shadows to track user interactions, as this most closely relates to my technical approach.

Starting with seminal work, there is Myron Krueger's Videoplace [145], which utilized user silhouettes for interactivity (though not true shadows). As far as I am aware, the earliest known work to leverage "shadow shape analysis" at the fingers for *touch input* is Andy Wilson's PlayAnywhere [308]. This system used a camera and illuminator that was fixed with respect to the input plane by virtue of the system being placed on a table. This system laid the conceptual groundwork for using shadows for detecting touch, measuring the decreasing distance between the shadow and fingertip.

More recent touch systems employing fixed cameras and fixed illuminators include ShadowReaching [263], Iacolina et al. [119], and Thomas [283]. Matsubara et al. [186] and Niikura et al. [208] used the same system, which featured two fixed infrared illuminators and a fixed camera capturing image pairs (one for each active illuminator). More ambitious is to have only a fixed camera and rely on existing natural or artificial light sources (i.e., no special illuminators). Adajania et al. [2], Paper Piano [288], ShadowSense [112], and Posner et al. [226] use this approach. However, with no control of the positioning between the camera and light sources (one or many, harsh or diffuse, bright or dim, etc.), input tends to be brittle, working well in some cases and failing in others.

Most similar to EclipseTouch, is ShadowTouch [159]. Unlike the above prior work, ShadowTouch is worn and mobile (i.e., not reliant on fixed external infrastructure). Like EclipseTouch, the system used an egocentric headset camera. Unlike EclipseTouch, ShadowTouch requires a special-purpose LED wristband. As there is no synchronization between the LED and camera, the LED is persistently lit, which is prohibitively energy consumptive for a wearable. Additionally, the authors note that they do "not have a special design to alleviate the ambient light interference" [159], an important part of my pipeline. I also move beyond ShadowTouch in terms of evaluation generalizability. In ShadowTouch's study, only three light-colored and matte surfaces are tested, only in a horizontal setting, and in one typically-lit environment. In this work, I test 12 surface materials (Figure 4.6), both dark and light, and matte and reflective. I also test vertical and horizontal orientations, and across three lighting conditions (typical lighting, bright, and dark). I note that my evaluation results show (Section 4.5) both systems to be of comparable accuracy; in other words, I achieve the same accuracy without the need for a special wearable.

4.3 Implementation

As a prototype platform, I instrumented a Meta Quest 3. In the following subsections, I step through the various hardware and software components that make up EclipseTouch.

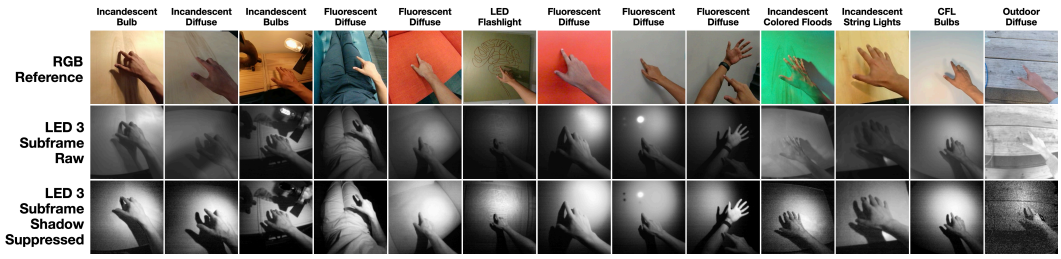


Figure 4.2: Examples of ambient shadow suppression output across a variety of materials, lighting types (incandescent, fluorescent tubes, CFL, LED, sun), and illumination conditions (diffuse, floodlight, point source). Note that shadows visible in visible light (RGB Reference) may be invisible in infrared and vice-versa. Figure 4.8 provides a step-by-step example of shadow suppression.

4.3.1 Camera

I affixed a HT-SUA33GM-T1V-C USB 3.0 camera [116] to the bottom left of my prototype headset (Figure 4.1). This global shutter camera has a modest resolution of 640x480 pixels, but is sufficient for my needs (i.e., my shadows are large and textureless, not requiring high detail). I fitted the camera with an 850 nm bandpass filter, matching the wavelength of my LED illuminators (discussed next). Even with this filter, other light sources can cast visible shadows at 850 nm, most notably the sun, as well as incandescent and halogen lights. To account for this, my pipeline includes an extraneous shadow suppression process, described in Section 4.3.5. As I rely on only a narrow frequency band of light, I purposely selected a monochrome camera with a wideband CMOS sensor (with no Bayer pattern, which would preclude imaging infrared light). Finally, this camera features an external hardware trigger, which I use to synchronize with my LED illuminators. The camera streams video over USB to a laptop, where my software runs. The maximum framerate of the camera is 791 FPS, though I operate it about half this speed to increase exposure time.

4.3.2 Illumination Geometry

As already discussed, EclipseTouch relies on active illumination to create structured shadows. I use LEDs with a wavelength of 850 nm (infrared), that are safe and invisible to humans, even in darkness. The LEDs are rated at 3 W, but I drive them at 1.1 W as a power conservation measure. Importantly, my prototype was built to serve as a vehicle for investigation. As such, I over-provisioned my headset with LEDs in each of the four corners. For the corner with the camera, the LED is placed directly above the camera. This arrangement can be seen in Figure 4.1. As I will evaluate and discuss later, some illuminator locations are more valuable than others, and so a commercial implementation would use fewer illuminators, potentially just one (as can be seen with my final prototype in Figure 4.13).

4.3.3 Driver Board

I use a Teensy 3.2 microcontroller and custom MOSFET LED driver board to precisely control the timing of my LED illuminators and camera frames (microsecond precision). My Teensy firmware uses the following five-step LED ”firing se-

quence": No LEDs on, LED 1 on, LED 2 on, LED 3 on, and LED 4 on. Each step in the sequence has a duration of 2.5 ms, and at each step the camera is triggered using its external pinouts. The firing sequence loops continuously, producing a 400 FPS raw video stream.

My camera uses an exposure time of 2.4 ms. I note that this is a comparatively short exposure time for a camera with a small sensor (1/5.6"), but this is not an issue for EclipseTouch because I am actively illuminating the scene. Furthermore, human skin is reflective in infrared, and the user's hands are never more than 70 cm away from the headset. As can be seen in Figures 4.2 and 4.6, the hand is readily seen, and the shadows cast are crisp and dark.

4.3.4 Video Stream

On my laptop receiving the 400 FPS camera stream, I read five frames at a time (i.e., a complete firing sequence), and composite this data into a new, singular frame containing multiple illumination sources: five 640x480 images side-by-side in a row. Thus, this new composited stream has a framerate of 80 FPS. The time between the start of the first frame (no illumination) and the end of the last frame (LED 4 on) is approximately 12.5 ms. This short duration means that even when the hands are in motion, the image can be stacked for image processing as though they were taken at essentially the same moment in time.

4.3.5 Extraneous Shadow Suppression

In addition to my infrared LEDs, other light sources with 850 nm wavelengths will cast finger shadows. Notable light sources include the sun, as well as some artificial lights, including incandescent and halogen bulbs (see example shadows in Figure 4.2). These shadows can generate false events, and so it is desirable to filter them out.

Importantly, light intensity is additive on image sensors (i.e., each CMOS pixel measures accumulated light, and if two or more light sources are contributing photons to this pixel, it will simply be the sum of intensities). I can use this property to great effect for removing unwanted shadows. Specifically, I can take my "no LEDs on" subframe, which captures any shadows generated from extraneous light sources, and simply subtract this from my other four subframes. This has the effect of removing the contribution of ambient light on those subframes, leaving only the illumination from that specific subframe's LED. An illustration of this process is shown in Figure 4.8, with example outputs across various environmental conditions seen in Figure 4.2.

4.3.6 Finger Tracking

With extraneous shadows removed, I next move to track the hands in front of the user. In an integrated system, this information would already be available from the hand tracking provided by the headset software. However, as I am using my own camera, I cannot simply use the Quest 3's hand tracking result, and instead must compute my own. For this, I utilize my LED 1 subframe, which provides an illuminated view of the hand with minimal shadows (as LED 1 is almost directly inline

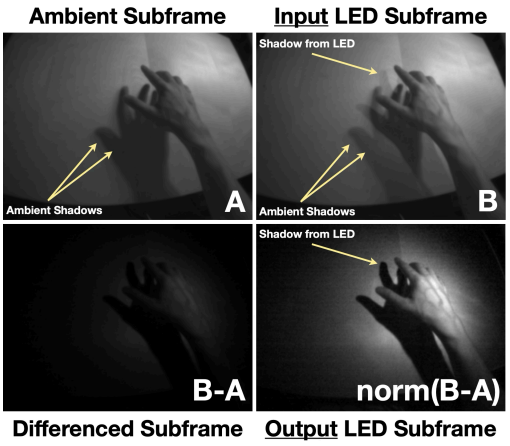


Figure 4.3: Overview of my shadow suppression process. My ambient subframe (A) captures shadows cast by ambient light sources. When an illuminator is on (LED 3 in this example), I cast a new additional shadow into the scene (B). Note this shadow can be weaker than ambient shadows, as is the case in this example (see also Figure 4.2). I then subtract the ambient frame from the LED-illuminated subframe (B-A). Having now subtracted ambient light, the frame becomes darker, and represents light only cast by the headset LED. To compensate for variable scene brightness (e.g., varying surface albedo and hand distance) I perform a final normalization — norm(B-A) — which accentuates the shadow. Note only a single shadow remains, the one cast by the headset LED.

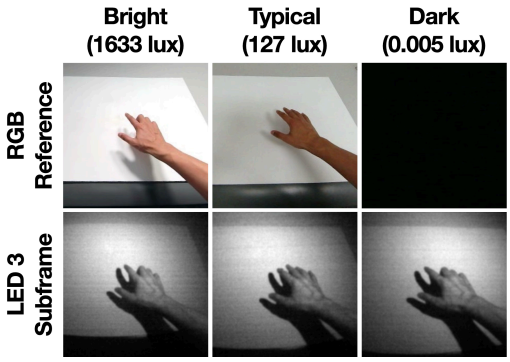


Figure 4.4: I tested three lighting conditions in my study: bright, typical, and dark. For reference only, I show the scene as it appears to a standard RGB camera along the top row. The bottom row shows the LED 3 illuminated subframe at the end of my pipeline (not shown are LED 1, 2 and 4 subframes, but the results are equivalent). In short, EclipseTouch is reasonably agnostic to ambient lighting condition as it provides its own illumination. Note also how shadows cast from ambient sources are not visible in the processed frame due to shadow suppression.



Figure 4.5: XR headsets on the market already include integrated infrared illuminators and infrared-sensitive cameras, suggesting EclipseTouch could be enabled via a software update. Additionally, I note that future AR glasses could also incorporate a single camera and LED in opposite corners of the frame, much like Ray-Ban Meta AI Glasses do today.

with the camera). I run Google's MediaPipe hand tracker [344], which provides 21 2.5D hand keypoints, though I only use the five fingertip points. I create 64x64 pixel patches centered on each fingertip. I use the wrist and Metacarpophalangeal (MCP) hand joints to normalize the size of the fingers (i.e., scale, irrespective of distance from the camera), and then the MCP and Proximal interphalangeal (PIP) finger joints to normalize the rotation (so that all fingers are pointing upwards in the patch). These finger patches are then passed to my ML model, described next.

4.3.7 Machine Learning

My deep learning model takes in as input a finger patch and a finger ID and jointly predicts touch state and hover distance. My model is a hybrid vision transformer, built on top of the FastViT T8 [285] backbone. Finger patches contain N channels ($N \in 1, 2, 3, 4$), one for each of the illuminator subframes. Finger patches are first passed through the backbone to produce image embeddings of size 768. These embeddings are then concatenated with a 5 dimensional one-hot encoded vector of the finger ID (thumb=1, pinky=5) to produce an embedding of size 773. Finally, this embedding is passed through a multi-layer perceptron (2 layers, hidden dimension 128, GeLU [106] activations) that encodes the scaled distance of the finger to the surface. Touch state is obtained by sigmoid activating and thresholding this value. I smooth touch and hover distance predictions with a mean filter over the 30 most recent frames.

4.3.8 Model Training Protocol

In my subsequent user study, I employ a leave-one-participant-out cross validation scheme to train and evaluate my models. My models were trained using the PyTorch and PyTorch Lightning deep learning frameworks. I initialized the FastViT backbone with ImageNet pretrained weights. I first trained the model to estimate touch state (by minimizing binary cross-entropy loss) and later fine-tuned the model to encode distance in the final logit (by minimizing the sum of mean absolute error and mean squared error). Models were trained for 10 epochs using the Adam optimizer, a batch size of 128 and a learning rate of 0.000003, which took about two hours on an NVIDIA 2080 Ti GPU.

4.3.9 Compute and Power Consumption

I carefully designed my model to be able to run as a lightweight background process, concurrent with numerous other models that already run on XR headsets (hand/body tracking, SLAM, etc). After training, I re-parameterize the model [285] to an equivalent one with fewer parameters (total 3.3M parameters). On an M2 MacBook Air — with similar hardware to the Apple Vision Pro — my model has an inference time of 0.47 ms. This means that my model can potentially run at ~2000 FPS, or run at e.g., 60 FPS consuming a small fraction of the headset's processing power.

As noted previously, my infrared LED illuminators consume 1.1 W when active. Only one LED is active at a time, and no LEDs are active 1/5th of the time, yielding a mean power draw of 0.9 W for illumination. My microcontroller and

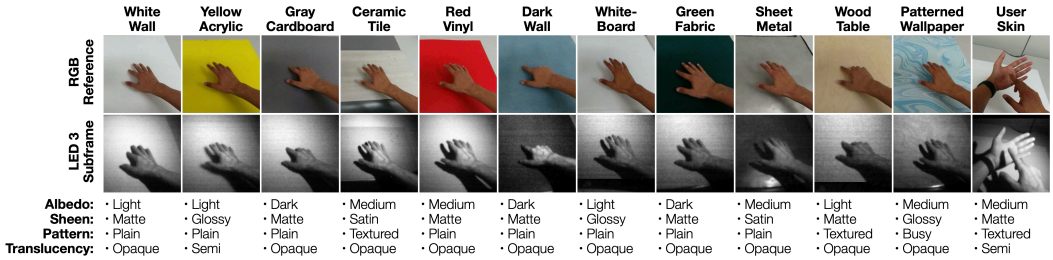


Figure 4.6: The twelve materials I tested in my study. Note the difference in appearance in RGB vs. infrared color spaces.

LED driver board consumes 0.3 W. My camera, running at 400 FPS, draws 0.8 W. As one reference point, the Meta Quest 3 draws ~8.6 W of power during use (giving its 18.9 Wh battery a stated 2.2 hours of runtime).

4.3.10 Compatibility with Existing Headsets

I note that several popular XR headsets already contain the requisite hardware to enable EclipseTouch. For example, the Apple Vision Pro contains two infrared illuminators and six monochromatic cameras sensitive to infrared light (Figure 4.5, left). The low-cost Meta Quest 3S contains two infrared illuminators and two monochromatic cameras (Figure 4.5, right). In both cases, the illuminators are used to boost hand tracking performance in low-light conditions [104]. However, their arrangement is also perfect for EclipseTouch — two infrared illuminators offset from one or more infrared cameras (more discussion in Section 4.5.1). Thus, it is likely that EclipseTouch could be enabled with a software update (at present, neither headset provides ad hoc surface touch segmentation, other than in a rough way using the native hand tracking).

4.4 Data Collection Protocol

To collect data to train and evaluate EclipseTouch, I recruited 10 participants (4 female, 6 male, all right-handed) for a one-hour user study. Participants were compensated $20 for their time. After completing consent paperwork, participants were fitted with my EclipseTouch-instrumented Quest 3, which streamed my 80 FPS video composite to a local desktop via USB where it was saved. The study was conducted in a windowless room with controlled lighting. I measured lux, reported later, using a Sper Scientific 840022 Light Meter. During the study, participants stood in front of a standing desk or a wall. Unlike prior work, I did not restrict participants' head motion or distance to surfaces.

I designed my study to capture a variety of conditions such that I could later analyze system performance across different materials (n=12, of varying albedo, sheen, patterns and translucency; Figure 4.6), surface orientations (horizontal and vertical), and lighting conditions (bright, typical, and dark; Figure 4.4). It was not possible to fully cross these conditions due to combinatorial explosion, and so I designed blocks of sessions to collect data, varying a single experimental factor at a time.

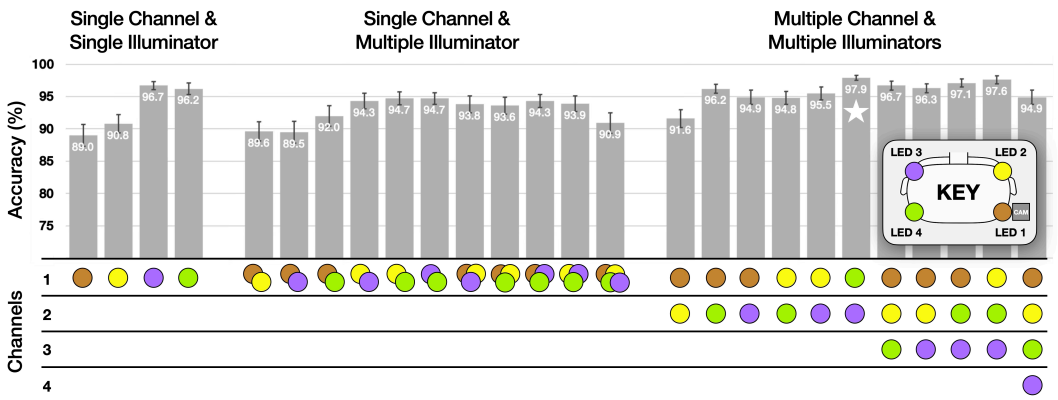


Figure 4.7: Results across illuminator combinations. A colored dot indicates that the corresponding illuminator is active in that channel. The best combination (LED 3 & LED 4, operating in separate channels) is marked with a star.

All of the individual sessions followed the same basic data collection procedure. First, participants were instructed to continuously touch and drag on a presented surface with their index finger, during which time 30 seconds of data was recorded. This was immediately followed by a second session capturing 30 seconds of data in which participants were asked to hover their finger above the surface and to perform repeated in-air taps. These trials provided positive and negative examples to train my machine learning model.

To collect data across a variety of surfaces, I curated a set of 11 diverse materials — including wood, plastic, metal, fabric, and painted/printed surfaces — seen in Figure 4.6 (along with a summary of their properties). These material samples were all cut down to 60×40 cm so as to be easily swapped and organized during the study. Both touch and hover sessions of data were collected for all 11 materials, in a random presentation order, in a horizontal orientation (placed on a desk), and in typical home lighting (measured at 127 lux, with typical home illumination around 100-200 lux [306]).

Next, I collected data in two additional lighting conditions: bright (1633 lux) and dark (0.005 lux). For reference, typical TV studio lighting is around 1000 lux and a quarter moon on a cloudless night is 0.01 lux [306]. For this study block, I reduced my material set down to two: white wall and patterned wallpaper. The former was a representative plain and light-colored material, and the latter served as a more challenging surface, being darker, glossy, and patterned. For these two materials, I collected two additional sessions of data in each lighting condition and in a horizontal orientation. Note in the previous block of sessions, I already captured data for these two materials under typical lighting (127 lux). For my three lighting conditions, I provide reference visible-light photos and infrared-illuminated subframes in Figure 4.4. I also collected data for surfaces in a vertical orientation. For this, I use my same pared-down material set — white wall and patterned wallpaper — and my typical home lighting condition.

Finally, the user's skin is a special, high-value input surface with unique optical properties (most notably subsurface scattering, creating diffuse shadows). In order to evaluate EclipseTouch's ability to extend to skin input (my 12th material), I collected eight sessions of data on the palm: two sessions each with the hand

held horizontal in my bright/typical/dark lighting conditions, and two sessions with the hand held vertically in typical lighting.

In total, each participant completed 42 sessions of data collection. With each session producing 30 seconds of data and a system framerate of 80 Hz, multiplied by 10 participants, this process yielded 1,008,000 individual frames for training and analysis.

4.5 Results and Discussion

My system prototype and study procedure was purposely designed to enable investigation of several important factors. First and foremost, I ran an ablation study to identify the most promising illuminator geometry, balancing accuracy with practicality. All subsequent results use this arrangement. Input to the skin is broken out as a special discussion. I conclude the section with two supplemental studies: hover distance estimation and multitouch. For touch prediction, I report classification accuracy and for hover distance estimation, I report mean absolute error in millimeters. Other than my illumination geometry ablation study (the next section), all results reported in this section are trained and tested with leave-one-participant-out cross validation, with no user or surface calibration. In all experiments, I always test on unseen users.

Across all materials (including skin), in both orientations and all three lighting conditions, EclipseTouch achieves an overall mean accuracy of 98.0% (SD=0.3%) using its best-performing LED 3 & 4 illumination geometry.

4.5.1 Across Illumination Geometries

First, I evaluate the performance of EclipseTouch under different illuminator configurations. My prototype hardware has four illuminators (Figure 4.1), which leads to 15 possible combinations ($2^4 - 1$) for illuminator placement on the headset. To enable triggerless operation and higher framerates, illuminators could also be turned on in groups of two, three, or four, and so I additionally ablate these 11 extra combinations. Due to the additive property of light transport, I can simulate these combinations by averaging the subframes from each of the LEDs. Figure 4.7 shows all the 26 illuminator combinations I tested. Prior to training each model, I modify the model architecture to accept images with different numbers of channels, by modifying the input channels of the first convolution layer of the backbone. I used six participants' data for training, and the remaining four participants' data for testing.

Results from this ablation study are shown in Figure 4.7. Focusing first on the Single Channel, Single Illuminator results, it is clear that LED illuminators 3 and 4 yield the best results (96.7% and 96.2% accuracy). Indeed, there is a clear pattern in performance relating to illuminator-camera distance. Figure 4.1 shows shadows formed by each of these illuminators. I note that illuminators 3 and 4 are the furthest offset from the camera and produce the most prominent shadows of the touching finger. Next farthest from the camera is illuminator 2, which produces notably smaller shadows of the touching finger. Finally illuminator 1, which is

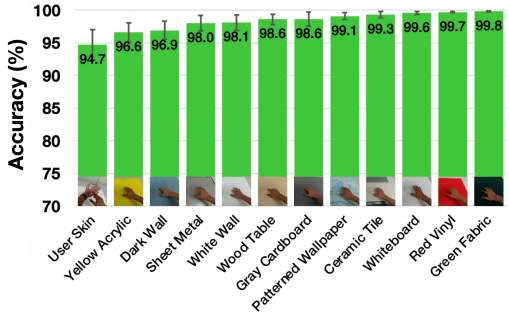


Figure 4.8: Touch classification accuracy vs. surface material.

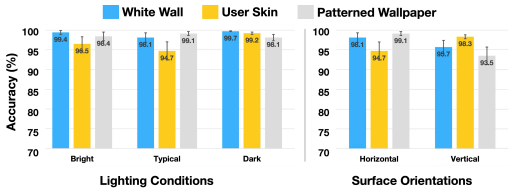


Figure 4.9: Touch classification accuracy vs. lighting condition (left) and surface orientation (right).

in-line with the camera, produces almost no shadow, leading to the worst performance of all the illuminators (89.0%).

Moving on to Single Channel, Multiple Illuminator configurations, where multiple illuminators are on at the same time, I observe that the best combinations are LED 3 + 4 (94.7%), as well as LED 2 + 4 (also 94.7%). While this performance is high, I observe that all combinations in this category consistently perform worse than their individual channel counterparts, as well as the best performing single-illuminator configurations. Simultaneously turning on multiple illuminators had no performance benefit over using one or more LEDs at optimal locations.

Finally, looking at Multiple Channel, Multiple Illuminator results, I find the best-performing combination is LED 3 & 4 at 97.9%, followed closely by LED 2, 3 & 4 at 97.6%. This makes sense, especially considering the fact that LED illuminators 3 and 4 already achieved high accuracies on their own.

Notably, the accuracy with even just one LED at an optimal location is similar to the best performing combination (96.7% for LED 3 alone vs. 97.9% for LEDs 3 & 4). This is promising, as existing headsets already include a pair of illuminators offset from integrated cameras (see Figure 4.5 and Section 4.3.10), potentially allowing EclipseTouch to be enabled through a software update. I also created a final prototype for demo purposes, seen in Figure 4.13, which features a single illuminator based on these results. For all subsequent study results, I report performance on this best-performing, multi-channel illuminator configuration: LED 3 & 4.

4.5.2 Across Surface Materials

A key component of my study was to evaluate the performance of EclipseTouch on a variety of everyday materials (see Figure 4.6 for the 11 non-skin materials that I tested). Importantly, these materials had a wide range of properties, including varying albedo, sheen, patterns and translucency. Results from this evaluation are shown in Figure 4.8. Across all materials tested, performance remains high, averaging 98.6% accuracy. This ranges from 96.6% for the yellow acrylic material to 99.8% for the green fabric material. Note that 9 out of 11 materials tested have accuracies above 98.0%, indicating strong generalization of EclipseTouch across surface materials.

Performance remains similar across albedo types, with light materials averaging 98.2% (SD=1.3%), medium materials averaging 99.0% (SD=0.7%), and dark

materials averaging 98.4% (SD=1.4%) accuracy. This is encouraging, since darker materials often pose a challenge to other vision-based systems.

Similarly, performance remains comparable across sheen types, with matte materials averaging 98.6% (SD=1.1%), satin materials averaging 98.7% (SD=0.9%), and glossy materials averaging 98.4% (SD=1.6%). Note that while I do include glossy materials, none of my materials were fully reflective, like glass. These materials do not produce shadows and thus cannot be supported.

Also encouraging is that performance remains high with materials with busy patterns; over 99.0% accuracy for my patterned wallpaper, wood table, and ceramic tile surfaces.

The worst performing material was the yellow acrylic, which had a mean accuracy of 96.6%. This material is semi-translucent, and produces shadows with softer edges as compared to the other materials in my study, potentially reducing performance. Opaque materials averaged 98.8% accuracy (SD=0.9%).

4.5.3 Across Lighting Conditions

Next, I analyze performance across lighting conditions. Figure 4.4 shows the three lighting conditions I tested in this study, ranging from dark (0.005 lux) to bright (1633 lux) lighting. Results for the three materials I studied across lighting conditions can be seen in Figure 4.9. On average, touch accuracies were 98.1% (SD=1.5%) for the bright condition, 97.3% (SD=2.3%) for the typical condition, and 99.0% (SD=0.8%) for the dark condition. Overall, performance remains similar across lighting conditions, with a slight increase in performance in the darkest lighting condition. This is an encouraging result, since most prior work either did not test or did not work well across such a wide range of lighting conditions.

4.5.4 Across Surface Orientations

Touch performance across the two orientations I tested can be seen in Figure 4.9. On average, performance was slightly better for the horizontal condition (97.3% accuracy, SD=2.3%) vs. the vertical condition (95.8% accuracy, SD=2.4%). Notably, the patterned wallpaper material had a sharp decrease in performance from 99.1% to 93.5% accuracy. I note that in my user study, participants typically had their hand closer to the headset in the vertical condition vs. the horizontal condition, which could have contributed to this decrease in performance. Furthermore, shadows of the touching finger look different in the vertical condition as compared to the horizontal condition, and since the majority of my training data was collected in the horizontal orientation, I hypothesize my models may be slightly biased towards this orientation.

4.5.5 On-Skin Touch Detection

An important surface material to study for touch input is the user's skin, as it is always available for input on the go. However, skin has unique properties that set it apart from the other materials I tested. It has varying albedo, surface texture, and some translucency (leading to subsurface scattering of light). It is also deformable

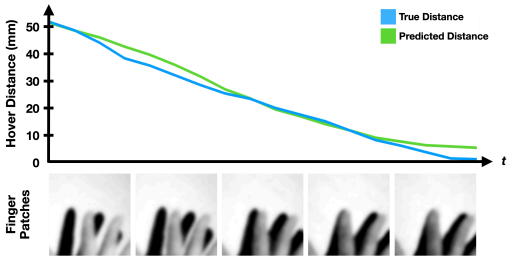


Figure 4.10: Example 250 ms sequence of an index finger descending to tap a surface. Note how the shadow evolves over time (illuminated by LED 3), converging towards the finger, and essentially disappearing upon contact. My model uses this visual information to predict hover distance.

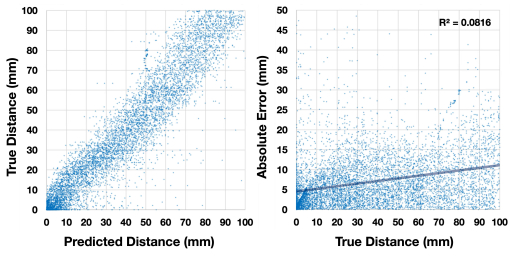


Figure 4.11: Left: Predicted vs. true hover distance. Overall, EclipseTouch is fairly linear and accurate, up to about 10 cm (mean error of 6.9 mm). Right: Mean absolute hover distance error vs. true distance — note the slight upward trend in error.

and non-planar. For this reason, I separately trained a model for skin input. I envision this model being used when the XR headset's existing hand tracking detects probable hand-to-skin input.

On-skin touch detection results are shown in Figures 4.8 and 4.9. Compared to other materials, detection accuracy was slightly lower, at 94.7% accuracy (SD=2.3%). I note, however, that this performance is still competitive with prior work [97,203]. Across lighting conditions, performance also remains high, with a noticeable jump in performance in the dark (99.2% accuracy). I note that prior vision-based on-skin touch systems, such as my own EgoTouch [203], have essentially 0% accuracy in the dark. Across orientations, performance is higher in the vertical orientation (98.3%) vs. the horizontal orientation (94.7%).

4.5.6 Supplemental Study: Hover Distance

As a user moves their finger towards a surface, the shadow cast by that finger moves and the distance between the fingertip and the tip of the shadow changes proportionally (Figure 4.10). For this reason, I hypothesized that it would be possible to train a model to estimate finger hover distance above a surface. To train and evaluate this model, I ran a small supplemental user study with 5 participants (2 male, 3 female). Participants filled out consent paperwork and were fitted with the EclipseTouch prototype. Participants stood in front of a table in a typically lit room. To collect ground truth distance of the finger from the surface, I affixed a HD USB webcam to the side of the table, so as to track the participant's touching finger from the side. At the start of the study, the experimenter calibrated the pixel displacements of the user's hand in the webcam view to real-world units (mm). Then, participants were asked to lift their index finger up and down above a certain location on the surface. Participants were instructed to only lift their finger vertically, so as to accurately track their fingertip's ground truth distance from the surface. I collected three sessions lasting 60 seconds each, for a total of 72,000 frames at 80 FPS.

To train this model, I simply fine-tuned my existing EclipseTouch model to directly estimate hover distance in the output touch classification logit (before

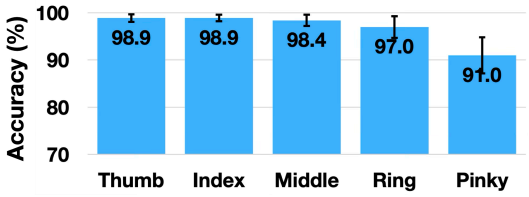


Figure 4.12: Results from my multitouch supplementary study, broken out by inputting finger.

activation). The main motivation for this design was that hover distance and touch classification confidence are highly correlated. Furthermore, this allowed me to have a single unified model that estimated both hover distance and touch state simultaneously. All models were trained with a leave-one-participant-out cross validation scheme.

Results from this supplemental study can be seen in Figure 4.11. In my hover distance region of interest, below 10 cm, my model had a mean absolute error of 6.9 mm. For ground truth distances less than 1 cm away from the surface, my model had a mean absolute error of 2.5 mm. Furthermore, I note that error increases slightly as distance from the surface increases (Figure 4.11). Example predictions of this model as a user touches a surface can be seen in Figure 4.10.

4.5.7 Supplemental Study: Multitouch

To evaluate the performance of EclipseTouch on multiple fingers, I ran a small supplemental study (in tandem with the hover distance estimation study). For this study, I collected four types of sessions, all with the white wall material in typical lighting. First, participants were asked to touch and drag across the surface with all their fingers simultaneously. Next, participants were asked to hover close to the surfaces and to perform in-air taps with all their fingers. Third, participants were asked to perform a pinch and zoom motion with their index and thumb fingers while touching the surface. Finally, participants were asked to perform the pinch and zoom motion in the air while hovering close to the surface. Each session lasted 30 seconds, and I collected three rounds per session, for a total of 144,000 frames at 80 FPS. I then fine-tuned my models with this data. Similar to before, I employed a leave-one-participant-out cross validation scheme to train my models.

Results from this study are shown in Figure 4.12. Similar to [159], I observe that the model performs similarly across the thumb, index and middle fingers (98.7% accuracy, SD=0.3%). Performance drops slightly for the ring finger (97%) and drops further for the pinky finger (91%), as they are sometimes not visible to the camera.

4.6 Limitations

While EclipseTouch represents a useful and practical increment over prior work, there is still room for improvement to achieve touchscreen-like performance. Foremost, similar to other vision-based approaches, EclipseTouch needs line-of-sight to the touching finger to function. On modern headsets, this issue has been par-



Figure 4.13: Based on my evaluation results, I created a final prototype featuring a single infrared emitter and camera.

tially addressed by incorporating multiple cameras on the headset, expanding the field of view outside that of the users.

I also note that EclipseTouch does not work across all surface materials. In particular, highly-reflective and transparent materials will fail, such as mirrors and glass (as no shadows are visible). In future work, the reflection of the finger on the surface itself could be used to estimate touch contact [218]. Apart from this, I also observed that very dark materials (in infrared) did not cast shadows, nor did highly 3D-textured surfaces (e.g., piled carpets, fur, fluffy clothing). Across all materials tested in my study, the two worst-performing were User Skin and Yellow Acrylic (the only two semi-translucent materials I tested). One hypothesis for this reduced performance is that subsurface scattering interferes with cast shadows. In the future, EclipseTouch could use its cameras and computer vision to identify unsuitable surfaces, and steer users to utilize compatible ones.

The power draw of EclipseTouch could be an obstacle to adoption in commercial systems (read more about the power consumption of my prototype in Section 4.3.9). In general, active illumination is power expensive. In systems using light emitters, such as the LIDAR and TrueDepth sensors in Apple's iPhones, the sensor is only turned on when needed. For example, the auto-unlock feature is first triggered by a motion event detected by a lower-power IMU, before turning on the more power-expensive TrueDepth sensor for Face ID. Even when on, the duty cycle is kept very low. A similar approach could be used for EclipseTouch. Contemporary XR headsets already build a model of the environment and track the user's hands for input. This data could be used to activate EclipseTouch opportunistically, when the hands are closer than e.g., 30 cm to a surface. When active, the infrared LEDs could be strobed for very short durations (my cameras are already externally triggered, so synchronization is not an issue), further reducing power consumption.

Another condition where EclipseTouch does not presently function is in direct sunlight. As noted in my study, my bright lighting condition was 1633 lux (exceeding that of typical TV studio lighting at 1000 lux [306]). It was certainly bright compared to a typical office, but not bright compared to the direct Sun, which can be 100,000 lux [306]. At this level of illumination, my 1.1 W LEDs cannot compete, and the shadows they cast simply disappear into noise. Other active illumination sensing methods have employed various strategies to combat this, including modulating and polarizing light, as well as using very narrow bandpass

optical filters. This is how, e.g., depth sensors such as Microsoft's Kinect and Apple's iPhone LIDAR can work in outdoor scenes.

4.7 Summary

In this chapter, I presented EclipseTouch, a headset-only system for detecting touches on everyday surfaces, including the user's skin, using worn infrared shadow casting. Moving beyond prior work, my results show that this approach is accurate (98.0% touch accuracy, 6.9 mm hover distance error), and works across a wide range of surface materials, lighting conditions and orientations, while running efficiently and with low latency. In addition, my ablation studies reveal optimal illuminator arrangement geometries and suggest that EclipseTouch could be implemented in existing headsets through a software update.

In this chapter I presented EclipseTouch and showed how ad hoc touch tracking can be enabled with the sensors already built into commercial headsets. I also discussed how touch can be an important implicit approach that can inform intent to interact with a surface or object. But one limitation of touch tracking is that although it senses interactions with the environment, it doesn't capture a lot of information about the user themselves. For the next two chapters, I will show how I attempted to get this information by using two different technical approaches for tracking the hand and body pose of a user.

Chapter 5

Arm Pose Estimation Using Commodity Smart Devices

This chapter was adapted from my published paper:

> Nathan DeVrio*, Vimal Mollyn*, Chris Harrison. 2023. SmartPoser: Arm Pose Estimation With a Smartphone and Smartwatch Using UWB and IMU Data. *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '23).*

5.1 Introduction

Despite being worn on the wrist, contemporary smartwatches know more about their users' legs, lungs, and heart than about their arms. For instance, smartwatches can predict locomotion mode, track total number of steps taken, and measure respiration and heart rate. Yet, beyond some very specific use cases (e.g., hand washing detection [11]), the activities of the arms and hands are unknown, despite the fact that they are the chief appendage we use to manipulate the world around us. If I could better track the arms, I could power more sophisticated applications

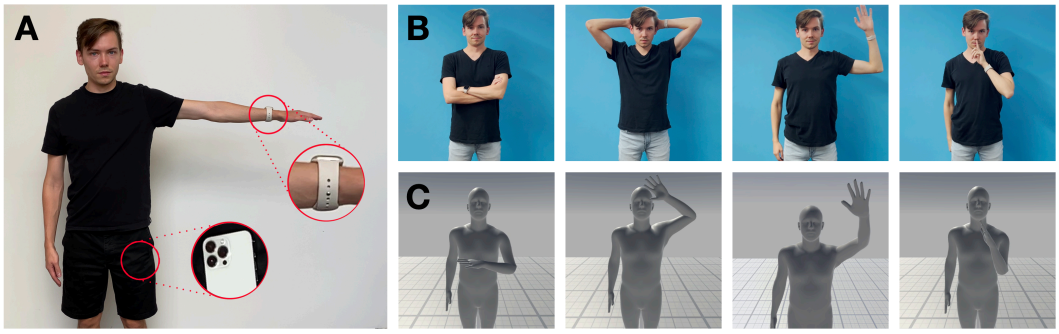


Figure 5.1: SmartPoser fuses UWB distance and IMU data from an off-the-shelf smartphone and smartwatch (A) to track a wearer's real-time arm pose (B & C).

spanning fitness [61,307], rehabilitation [21], life logging [195], occupational training [35], and context-aware assistants [233], to name just a few domains.

Researchers have long recognized the importance of tracking arm pose to support such use cases. As I will discuss in the Related Work section, a wide variety of methods have been brought to bear on this problem. Much prior work has utilized external, fixed equipment (most often cameras), but this immediately precludes mobile use. Next most common are worn sensor arrays or suits, which are comparatively more practical, but still face significant consumer adoption headwinds. Ideally, I could track a user's arm pose with devices they already own and routinely carry with them [120, 234], relying on no external infrastructure.

In this work, I present SmartPoser, a new arm pose tracking method that uses only an off-the-shelf smartwatch and smartphone (Figure 5.1). In other words, my technique is software only, and could be enabled on recent devices as an over-the-air update or app download. SmartPoser also allows users to be on the move while it performs its tracking, a capability not found in many prior approaches [254,350]. Taken together, these advances could significantly improve the ease-of-use and practicality of mobile arm-tracking applications.

Uniquely, I take advantage of recent ultra-wideband (UWB) functionality to capture the distance between the smartwatch and smartphone. I fuse this with more conventional IMU-reported data, specifically watch and phone orientation and acceleration. Importantly, the UWB measurements are absolute, and less drift-prone as compared to relative inertial data used in most prior work. As a consequence, my tracking output is more stable and accurate. In my user study, my system achieved a median positional error of 11.0 cm for the wrist and elbow joints, which is comparable to prior work requiring uncommon hardware.

The contributions of this work are as follows: 1) A self-contained and software-only approach for arm pose tracking combining UWB and IMU data, which is unique in the literature. 2) A functional, real-time implementation using two common off-the-shelf devices. 3) User studies that show my system achieves the best tracking accuracy among systems that do not require special user instrumentation or external infrastructure. 4) Open-sourced, synchronized UWB/IMU/Kinect data, processing pipeline, and trained models.

5.2 Related Work

In this section, I review past systems for arm pose tracking. In addition to arm tracking, I also discuss systems for whole-body pose tracking that have used comparable sensing technologies. I first cover systems that use UWB for pose tracking, before moving to IMU-based systems. I then conclude with systems most similar to my own — ones that combine inertial data with some form of distance-based measurement.

5.2.1 UWB-Based Pose Tracking

My system takes advantage of ultra-wideband (UWB) ranging. This technique uses the time-of-flight of a challenge-response transmission to determine the distance between two devices. Unlike other more popular RF technologies, such as WiFi

or Bluetooth, UWB operates at a very low energy level and instead of varying frequency or power, the technology transmits information at specific times using a large bandwidth. This results in multiple beneficial properties such as high immunity to multipath fading, high data throughput, and excellent time-domain resolution, allowing for more accurate localization and tracking, down to centimeters [88].

Due to the strengths just described, UWB has become a popular approach for pose tracking systems due to its ability to accurately measure the distance between devices in space. A common setup is to place UWB beacons on the limbs and torso of a user and instrument an environment with one or more receivers to measure the time-of-flight of signals transmitted from the beacons [191, 328]. Researchers have also created a UWB radar with a large, MIMO antenna array that can capture a radio "image" of the user's pose [265]. When compared to cameras, UWB can be more privacy-sensitive and can operate through walls and other occlusions [265].

Beyond UWB, there have also been other RF technologies used for pose tracking, many of which share similar advantages to those just described for UWB. In comparable setups, researchers have placed a transmitter in the environment and tracked the pose of a user (often uninstrumented) by measuring signal reflections. Examples of such systems include ones that use RFID scanners and tags [133, 295], mmWave radar [249], and even WiFi [237, 238]. One drawback of systems that use purely RF ranging is that although they are able to measure the distance to different parts of the body, it is difficult to distinguish fine limb orientations such as the rotation of the wrist. For this reason, RF systems are often combined with IMU sensors that can provide information on limb orientation [51, 108, 353].

5.2.2 IMU-Based Wearable Pose Tracking

The idea of tracking pose with a wearable system of inertial sensors did not surface until the miniaturization of magnetic, angular-rate, and gravity (MARG) sensors in the late 1990s [19]. Pioneering systems had to deal with the problem of fusing measurements from multiple instruments to track limbs, all the while minimizing drift. In addition, these systems also had difficulty detecting and correcting for erroneous measurements such as magnetic disturbances [19, 183]. The two dominant approaches that were developed to solve these problems include Complimentary filters by Bachman et al. [19] and Kalman filters by Marins et al. [183]. While these two seminal works would develop the theoretical foundations for tracking pose with inertial data, many others went on to take these ideas and develop them into real-world systems [168, 197, 337, 351]. The commercial Xsens system can be seen as a current peak of this classical filtering and optimization-based approach [240]. By using a suit embedded with 17 IMUs located all over the body, this system is able to track a wide range of motions at 120 FPS [240].

In recent years, a new class of systems has emerged which are able to use significantly fewer IMUs to track pose with a similar level of accuracy [38, 131, 253, 254, 301, 332, 333]. A breakthrough and common theme in these systems has been modeling inertial data over multiple frames in time, often with neural networks. A seminal system is Sparse Inertial Poser (SIP) by Marcard et al., which was able to drop the sensor burden from 17 IMUs down to 6 by using an optimization-based method [290]. Building upon this, Huang et al.'s Deep Inertial Poser (DIP) [115]

removed the constraint in SIP of needing to access the entire sequence of data and instead only used a subset of nearby frames in a bidirectional RNN (Bi-RNN). More recently, TransPose [333] and Physical Inertial Poser (PIP) [332] advanced DIP's Bi-RNN approach, improving fidelity and framerate (to 90 FPS). This was primarily achieved by decomposing the model into sub-tasks that are easier to solve and combining the outputs to get user pose and global translation. More recently, IMUPoser [201] used IMUs already found in smartphones, smartwatches and earbuds to produce a best-guess pose from one to three points of instrumentation.

Of the IMU-based systems, the most similar to ours is ArmTrak, developed by Shen et al. [254]. Similar to SmartPoser, the system focused on tracking arm pose (instead of full-body pose) and did this by using a single IMU on the wrist instead of multiple IMUs placed across the body. Unlike the later deep learning methods, they used a hidden Markov model to probabilistically estimate next states for arm joint positions. This approach produced good arm pose estimates using a single IMU. However, the method was slow (with reduced accuracy when run in real-time) and could not track users on the move (only evaluated with feet planted after calibration). I detail my improvements over ArmTrak, particularly tracking results, in subsection 5.5.5.

5.2.3 Fusing Inertial and Distance Measurements for Wearable Pose Tracking

One of the greatest challenges in working with IMUs, particularly consumer-grade components, is noise and drift. For this reason, it is often advantageous to have a secondary sensor stream that can provide absolute position or orientation data. In the context of body tracking, worn systems have combined IMU data with magnetic field sensors [241, 312], ultrasonic transmitters and microphones [129, 169, 289], infrared distance sensors [319], micro-flow sensors [171], acoustic ranging [350], and optical linear encoders [207]. As previously mentioned, UWB beacons have also been used for this purpose, albeit not in a wearable form factor [51, 108, 353]. For each of these sensing approaches, there are important trade-offs to consider such as cost, robustness to occlusion, and across-environment reliability. Yet despite their differences, these sensors all share the same operating principle of capturing absolute measurements to augment the relative measurements of worn IMUs. Most importantly, all of the aforementioned systems in this section require new and special devices to be worn by the user. A key contribution of this work is showing this ranging+IMU fusion approach can now be achieved on popular commodity devices that users already own.

5.3 Implementation

My motivation in developing SmartPoser was to track arm pose using only a smartphone and smartwatch (Figure 5.1). Figure 5.2 provides a high-level system overview and data flow. I now describe the core components of my system.

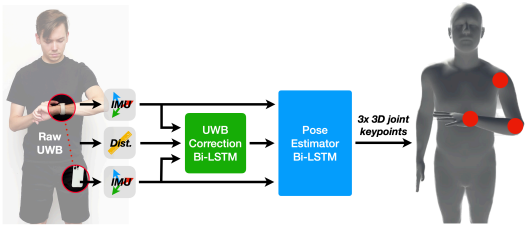


Figure 5.2: SmartPoser system overview. My model has 46.4k parameters in total, with 3k parameters for the UWB Correction model and 43.4k parameters for the Pose Estimator model.

5.3.1 Proof-of-Concept Devices

I selected an iPhone 13 Pro (iOS 16.4) and an Apple Watch Series 7 (watchOS 9.4) as a pair of example, off-the-shelf, and popular smart devices. This choice was also motivated by Apple's mature UWB and inter-device communication APIs that facilitated my development. Android lags behind in this regard, with Google adding UWB support via a Jetpack library only in late 2022 [84]. This is to support applications like lost object tracking, unlocking cars, and AR localization. Fortunately, the number of UWB-capable smart devices has steadily grown in recent years [305], and the feature looks to become increasingly pervasive.

5.3.2 UWB Ranging & IMU Data

In my custom app, I use the iOS Nearby Interaction API [10] to set up a UWB ranging session between the watch and phone. Once the session has been established, I receive distance updates at approximately 5 Hz. I note that iPhones offer higher sampling rates and even azimuth and elevation readings when a UWB device is located within a narrow cone-shaped field-of-view projecting from the rear of the device. Unfortunately, the field of view is too narrow for me to make practical use of this capability.

To get data from the IMUs onboard the phone and watch, I used Apple's Core Motion API [9], which is standard across all iOS devices. Using the API, I acquire data about the current orientation quaternions and acceleration measurements of each device at roughly 25 FPS. I use this as the native frame rate of my system, upsampling the 5 Hz UWB data to the more responsive 25 Hz IMU rate by duplicating the most recent UWB value. Finally, I note that although the two devices can estimate the gravity vector and magnetic north, I found it was more reliable to explicitly align the two devices' frames of reference. For this, I require the devices to be placed side by side, which could happen e.g., in a charging dock overnight.

5.3.3 UWB Correction

In early piloting, I saw that UWB measurements exhibited systematic offsets and characteristic noise, especially for certain arm pose regions. This is chiefly due to loss of line-of-sight between the two devices due to body occlusion (e.g., putting the arms behind the head). The example data in Figure 5.3 illustrates this effect. To capture this data, the authors placed the iPhone 13 Pro smartphone in their

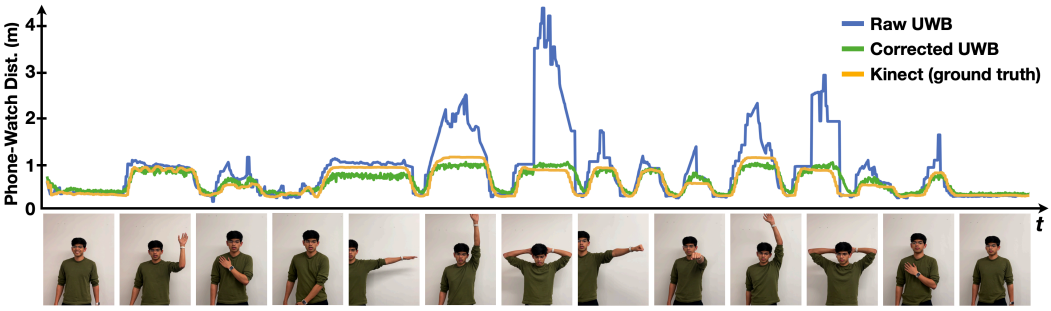


Figure 5.3: Example user motion sequence over one minute. Raw UWB distance is plotted in blue — note the significant errors, far exceeding possible arm lengths, when the watch loses line-of-sight to the phone. My corrected UWB stream is plotted in green. Ground truth distance provided by an external Azure Kinect is shown in orange.

left-front pants pocket, and wore the Apple Watch Series 7 smartwatch on their left wrist. The authors then performed a range of exemplary arm motions. During this process, the phone-to-watch distance was recorded (Figure 5.3, blue line). To provide a ground truth distance, I used an Azure Kinect and its native SDK to capture a 3D skeleton of the user in real-world units. Using this skeleton, I "attached" a virtual phone and watch, and compute the distance between the two (Figure 5.3, orange line).

Given the offset and noise is semi-structured, I can compensate for some error, especially if I take advantage of orientation data from the IMUs, which already provide some cues as to the orientation and location of the arms. For this, I designed a UWB correction network, which I implemented as a 2-layer bi-directional LSTM with 8 hidden units each. The input to the RNN is a 125-long sequence (5 seconds of data at my system's native framerate of 25 FPS) of raw UWB distance measurements, plus phone and watch orientations (3x3 rotation matrix format) and accelerations (X/Y/Z), for a total feature dimension of 25. I scale the UWB distance measurements by the wearer's arm span. The scaling ensures that the model is able to generalize to various body sizes. I also scale the acceleration values by 30 in order to make them suitable for training, following prior work [332, 333]. The RNN outputs a sequence of corrected distance measurements. The correcting effect of this RNN is illustrated in Figure 5.3, green line. In this example sequence, mean UWB tracking error is reduced from 34.5 cm to 6.7 cm, and is considerably more stable as well.

5.3.4 Pose Estimation

My pose estimation RNN ingests corrected UWB data along with raw IMU data (watch and phone orientation and acceleration). Taking inspiration from Deep Inertial Poser [115] and TransPose [333], this network is also implemented as a 2-layer bi-directional LSTM, with 32 hidden units each. Watch orientation and acceleration are normalized such that they are relative to the phone's frame of reference. Along with having an absolute distance measurement, using relative orientations helps to make the system more robust against IMU drift. As with UWB correction, I scale the acceleration values by 30. I use the same input vector

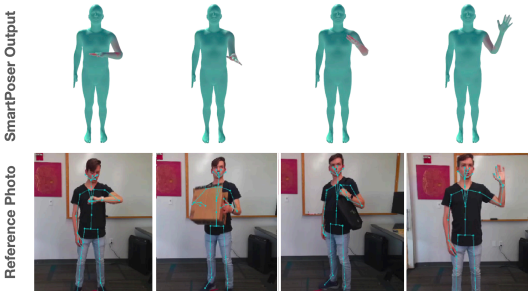


Figure 5.4: Example set of predicted poses. Vertex error (vs. Kinect ground truth) is color coded.



Figure 5.5: The 20 terminal arm poses utilized in my user study. Note that participants were asked to transition between these poses and all intermediate data was captured for evaluation (i.e., not just the terminal poses).

format as my UWB correction network – a 125-length sequence of UWB distances, phone and watch orientations and accelerations. This 125-length sequence is a rolling input buffer, and new predictions are made in real-time whenever a new sample arrives in the buffer (at 25 Hz).

This RNN outputs a 125-long arm pose sequence, containing 3D joint locations for the wrist, elbow, and shoulder. The predicted joint locations are 9 values normalized to the user's arm span, which I can "reproject" into real-world units by scaling them by the user's arm span. Rather than taking the most recent pose prediction (i.e., the 125th frame), I use the 120th frame. Functionally, this means I am predicting 200 ms behind real-time. This approach to use the n-5th output frame was directly inspired by TransPose [333], and offers a compromise between latency vs. accuracy. Note, during testing I also used the n-5th frame of ground truth to match with my prediction frame.

5.3.5 Model Training

In summary, SmartPoser consists of two machine learning elements – UWB correction and then pose estimation. I train the full system end-to-end using the Adam optimizer, with a learning rate of 3e−4. The UWB corrector is trained to regress to the Kinect-derived distances (Section 5.3.3), scaled by user arm span, and using mean squared error loss. The pose estimator network is trained to regress to arm joint positions (shoulder, elbow, and wrist) using a mean-per-joint-position-

error (MPJPE) loss. Kinect-derived ground truth arm joint positions are preprocessed to have a shoulder frame of reference (i.e., ignoring world body rotation).

The total loss of my system is the sum of the UWB corrector loss plus the pose estimator loss. The system is implemented using PyTorch and the PyTorch Lightning frameworks. For efficient training, I train the model to predict a full motion window of 125 samples (5 seconds) rather than just the last frame, following the method in [115, 333]. In other words, during training I use non-overlapping windows, however for testing I overlap windows by all but one new frame.

5.3.6 Model Inference and SmartPoser Output

To facilitate development and debugging, UWB and IMU data is streamed to a laptop (M1 MacBook Air 2021). On this machine, my trained model takes 8.1 ms per inference (on the CPU, using pytorch), meaning I output pose predictions as fast as data arrives (25 FPS). To verify SmartPoser could run entirely on-device, I also converted my trained model to CoreML with mixed precision compute and found it took only 0.99 ms per inference on an iPhone 12 Pro. Although I envision my arm pose data most often being used internally and unseen (discussion in Section 5.6), I did build a basic visualization tool, seen Figures 1, 5.2, and 5.4.

5.4 Evaluation

To evaluate the performance of SmartPoser, I conducted a user study with 10 participants (mean age 25.4; 3 identified as female, 7 as male). The study took place in a typical indoor office environment, lasted approximately 45 minutes, and compensated $20. The study was divided into two data collection phases. The first phase focused on arm motions a user might perform in the course of their daily activities. Phase two captured less ecologically valid, but more varied and challenging gross motor arm poses (Figure 6.9).

Before participants wore my test devices, the smartphone and smartwatch were placed side-by-side, as they might rest in a charging dock. I use this to align their independent reference frames into a common, global frame of reference. I then asked participants to wear the smartwatch on their left wrist and to place the smartphone in their left front pocket (see Figure 5.1), which is a common placement [120, 234] (though I note other arrangements are possible, which I discuss in Limitations). Before each study phase began, I had users perform a T-pose with their arms (palms facing down, such that the watch face is parallel to the ground) in order to capture the relative orientations of the two devices on the body (and importantly, the same study procedure used in IMUPoser [201], Transpose [333], Deep Inertial Poser [115], and Physical Inertial Poser [332]).

5.4.1 Daily Activities Procedure

The first phase of data collection focused on arm poses that users might assume during daily routines. To capture such data, I had participants complete an "obstacle course" style study (see e.g., [23, 122]). I selected 20 brief mock activities, setup at stations within my study environment. These activities were divided into

seven categories: *Domestic* (ironing, open jar, vacuum floor, wipe table), *Grooming* (comb hair, wipe forehead, wash with loofa), *Food & Beverage* (stir pot, pour teapot, drink from cup), *Moving Items* (move grocery bag, lift box above head, wear bag on shoulder), *Exercise* (jumping jacks, biceps curls), *Home DIY* (use screwdriver, use paint roller on wall), and *Miscellaneous* (check time on watch, read magazine, flick light switch). Although this set of activities is not comprehensive, I have confidence my system did not overfit. For example, the Demo Activities segment of my Video Figure shows predictions of unseen poses resulting from naturalistic activities. This is a good signal of expected performance for poses and activities not used in training and matches my anecdotal observations throughout the course of development.

A large TV screen was positioned in the study area with a list of the activities. In addition to this visual prompt, the experimenter also spoke aloud the next activity to perform, and controlled the visual display. Although some activities were bimanual, in general I asked participants to use their left arm, as it was the one being tracked. All activities were performed one after another while data was continuously recorded, thus beyond the activities themselves, I also captured the transitions between them and miscellaneous organic gestures such as participants adjusting their clothes. Participants completed this procedure twice, taking roughly 10-15 minutes in total.

5.4.2 Gross Motor Arm Pose Procedure

Real-world activities tend to be performed in front of the body, or at least close to the torso. As such, they tend to have less variance and less extreme arm positions. Thus, as a compliment to my phase one data collection, I also captured a more gross-motor-oriented arm pose dataset. For this, I use a set of 20 terminal arm poses, seen in Figure 6.9. These exemplify a variety of spatial endpoints, joint positions, and joint orientations.

As with phase one, I used a TV screen to prompt participants with a random sequence of terminal poses (20 poses, 10 repeats, random order). Importantly, data was continuously captured as participants transitioned between these poses. Thus, instead of capturing 20 poses, my study captures transitions between ~$_{20}C_2$ pose pairs, so the effective variety and volume of motions I captured was large. In addition, the amount of time spent in the terminal poses is very low, avoiding bias. Transitioning through the 200-pose sequence took approximately 10-15 minutes.

5.4.3 Data Capture & Ground Truth

Throughout both phases of the study, I used an Azure Kinect [281] to capture ground truth data on joint positions and orientations of participants' arms. The Kinect ran continuously, and whenever I got a new frame of UWB+IMU sensor data (at ~25 Hz), I paired it with the most recent Kinect frame. Across all participants, my activities dataset contained 151k frames of data (representing roughly 105 minutes of data), and my gross motor dataset contained 129K frames of data (~90 minutes).



Figure 5.6: Median per-joint positional error was 11.0 cm averaged across all study participants for the elbow and wrist joints. For the gross motions phase, median per-joint positional error was 11.2 cm, 9.2 cm for just the elbow, and 13.2 cm for the wrist. For the real-world activities phase, median per-joint position error was 11.1 cm, 8.4 cm for just the elbow, and 13.7 cm for the wrist. Error bars are standard deviation (cm).

Joint	Elbow		Wrist		Mean	
Cross-user metric (cm)	Mean	STD	Mean	STD	Mean	STD
Median error	8.7	0.9	13.3	1.0	11.0	0.9
Mean per-joint positional error	10.0	0.9	15.1	1.2	12.6	1.1
Mean absolute error	15.0	1.4	22.4	1.7	18.7	1.5

Figure 5.7: Calculated metrics for all arm joints for the combined data set of gross motions and daily activities. All metrics are reported in cm.

5.5 Results & Discussion

5.5.1 Train/Test Procedure

I evaluated my model using a leave-one-participant-out cross validation scheme (i.e., ten-fold cross validation with whole participants' data as folds). This simulates "out-of-the-box" accuracy that requires only a single calibratory pose and no training data from the wearer (whereas much prior work requires per-user training data, or even per-worn-session training data). While I train on real-world activities and gross motor arm motions datasets combined, I break these results out where appropriate to give further insight into real-world accuracy.

5.5.2 Pose Tracking Accuracy

Across all participants, I found a median error (averaging elbow and wrist joints) of 11.0 cm (SD=0.9 cm). As I move along the kinematic chain, error unsurprisingly increases — median error by joint: shoulder 0.17 cm (SD=0.02 cm), elbow 8.7 cm (SD=0.9 cm), and wrist 13.3 cm (SD=0.98 cm). Looking at error broken out by gross motor arm poses vs. real-world activities, I find median error (elbow and wrist) of 11.2 cm (SD=2.3 cm) and 11.1 cm (SD=1.3 cm) respectively. Figure 5.6 provides a summary of these results for median error. Other results metrics for the combined data set of gross motions and real-world activities can be found in Table 5.7, and the calculated CDFs in Figure 5.9. My results already used a leave-one-user-out cross validation and had a low STD of 0.9 cm across users. However, to further verify I was not overfitting, I also computed metrics testing on the training set specifically and observed a median wrist error of 11.9 cm (vs. 13.3 cm from cross validation).

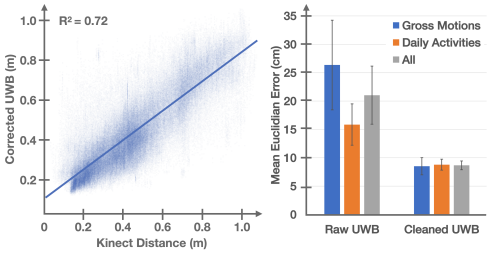


Figure 5.8: Left: Plot of corrected UWB distance vs. Kinect-derived distance (280k datapoints). Right: UWB Correction RNN performance across study datasets. Mean Euclidean error reduces from 21.0 cm to 8.6 cm on average. Error bars are standard deviation.

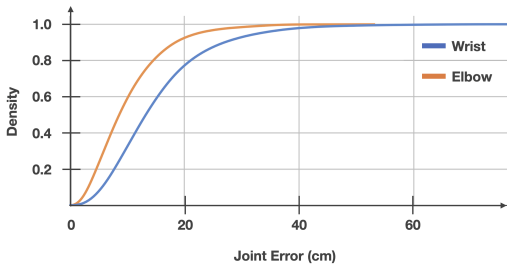


Figure 5.9: Cumulative distribution functions (CDF) of the elbow and wrist joint errors.

5.5.3 Spatial Error

As discussed earlier, UWB ranging is sensitive to occlusion (in my case, body occlusion). My UWB Correction RNN (Section 5.3.3) compensates for some of this error, but not entirely. Figure 5.8 compares mean euclidean error of the raw UWB distances (21.0 cm) vs. My corrected UWB values (8.6 cm), broken out by study dataset. I also generated a scatter plot (Figure 5.8) plotting my corrected UWB distance vs. the Kinect-derived ground truth distance for all 131k frames of study data from my gross arm pose procedure (representing ~1.5 hours of data). I include a line of best fit, which has an $R^2 = 0.72$.

Although the correlation is strong, there are other error modes and behaviors visible. To further investigate this, I created a series of heatmaps visualizing the maximum UWB error in space around the body using my study data. I look at arm pose data from all participants combined, and then only the wrist joint, which had the largest addressable volume and largest median error.

Figure 5.10 provides three heatmaps – a frontal, side, and top-down view of a wearer. Heatmaps are shown for the error of raw UWB distance, corrected UWB distance, and joint predictions versus Kinect-derived ground truth. All the heatmaps have some common error hot spots (red-orange regions in Figure 5.10), while the volume in the front and to the left side of the body is reasonably accurate. Most of the regions show up in blue, corresponding to an error of between 0-15 cm. Wrist locations that could lead to potential self-occlusion, such as the right thigh and the space above the left shoulder, intuitively show up as highly erroneous regions in the raw UWB distance heatmaps. Note how my UWB correction module is able to account for most of these errors. The error-prone areas for the wrist joint error map well with the error-prone regions for corrected UWB distance and are altogether much lower than the original errors for raw UWB distance.

5.5.4 UWB Ablation Study

The primary contribution of SmartPoser is the introduction of UWB distance data to improve arm tracking results. To quantify how much the improvement was, I ran an ablation study of my complete UWB+IMU system versus one that used only IMU data. The results of this study can be seen in Figure 5.11.

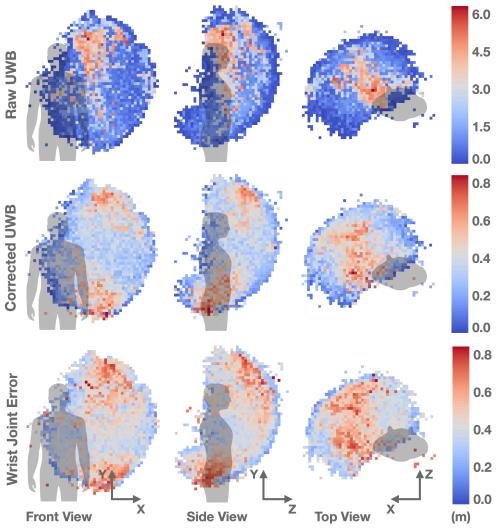


Figure 5.10: Three rows of heatmaps visualizing maximum absolute Euclidean distance error for raw UWB distances (i.e., not corrected), corrected UWB distances, and predicted wrist joint position. For all measurements, error (in meters) is calculated with respect to Kinect-derived ground truth. Note the scale for the raw UWB distances (top row) is different from the bottom two rows.

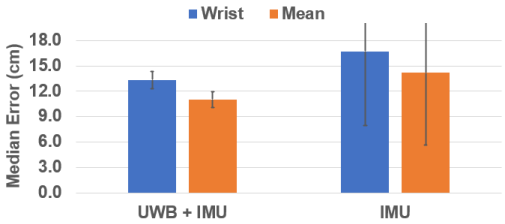


Figure 5.11: Ablation results showing the effectiveness of using both UWB and IMU data for pose estimation. Results are calculated on data from both phases of the user study. When adding UWB data to the model trained on IMU alone, median error for both joints decreases from 14.2 cm to 11.0 cm.

Averaging the results from both the gross motions and daily activities phases, I observed a median error of 11.7 cm for the elbow and 16.7 cm for the wrist for the model with only IMU data. Compared to my standard results from the model with UWB+IMU, this meant error was 3.0 cm (25.6%) greater for the elbow and 3.4 cm (20.4%) greater for the wrist when using only IMU. Additionally, when looking at pose outputs, I find UWB+IMU is much more stable. For example, when holding static poses with IMUs alone they tend to ”droop” slowly to a mean pose. These results show that the contribution of UWB distance measurements is significant to SmartPoser and a key to helping enable higher-fidelity pose tracking.

5.5.5 Comparison to Prior Work

To contextualize my results, I compare SmartPoser accuracy to other arm tracking systems that also do not require special instrumentation (i.e., no new or exotic sensors worn on the body) or external infrastructure (i.e., entirely self-reliant and work on-the-go). While these systems were evaluated on different datasets, I believe it is applicable to compare them to SmartPoser.

ArmTrak [254] achieved a real-time median error of 12.0 cm for elbow and 13.3 cm for wrist while performing free motions with the arm (accuracy was higher when using an offline model). Of note, ArmTrak required participants to stand still (i.e., feet planted without the ability to change body orientation), unlike SmartPoser. IMUPoser [201] did not report median error, but achieved a combined real-time mean error of 21.6 cm for all arm joints and 26.9 cm for just the wrist.

MUSE [253] only estimated position at the wrist for 5 participants and achieved a real-time (10 Hz on a desktop computer) median error of 13.0 cm for motions when the body moves around. Compared to these works, my system achieved a real-time (25 Hz) median error of 11.0 cm for the shoulder and wrist, and 13.3 cm for just the wrist.

5.6 Example Uses

Arm pose tracking (and body pose more broadly) has been well-motivated in prior work. My system does not enable any new applications per se, but rather makes them more practical — able to run on off-the-shelf consumer hardware many people already own. For this reason, I do not re-implement any application demos, and instead point to the literature for motivating use cases. Please see my Video Figure for real-time demos.

Fitness - Arms are involved in many exercises, and thus tracking the arms has obvious and immediate utility for fitness-oriented applications. The most straightforward are applications that automatically track rep counts for lifting free weights, jumping jacks, jump rope, and similar activities. One could also imagine a smartwatch assistant verbally guiding a user through a tai chi routine, automatically advancing steps. It may even be possible to track exercise form for personalized feedback; for example, proper extension of the arm when hitting a punching bag or form when shooting a basketball.

Rehabilitation - Tracking of the arms could also prove useful in rehabilitation (e.g., rotor cuff injuries). For example, continuous background tracking could supplement periodic sessions with physical and occupational therapists by monitoring arm metrics such as total motion, duration lifted, maximum extension, and maximum height reached. It could even be that appointments are automatically scheduled if there is limited progress or performance loss in a given metric. Alternatively, instead of passive, background monitoring, arm tracking could be employed in interactive rehabilitation applications. For example, an app could guide users through a series of balance exercises or reaching tasks. Such apps could not only automatically advance steps, but provide instructive feedback and encouragement.

Life Logging - Arm tracking could allow for new types of activities to be automatically tracked for life logging purposes. For example, it might be possible to track cigarette consumption (repeatedly raising one's hands to the mouth with a particular cadence) as part of a smoking cessation effort. Likewise, detecting periods of food consumption may also be possible for dieting apps. It may also be possible to log domestic activities, such as cooking, cleaning, vacuuming, folding clothes, and putting away dishes.

Occupational Safety & Training - Many forms of manual labor employ the hands, such as carrying boxes, stacking groceries, picking vegetables, driving a truck, laying bricks, mowing lawns, painting a house, cleaning windows, etc., which could be tracked to ensure workers are not fatigued, exploited or injured, e.g., Repetitive Strain Injury (RSI) [105], Hand-Arm Vibration Syndrome (HAVS) [255]. It may even be possible to detect improper handling of power tools or heavy

equipment to improve site safety. Training on how to properly handle a tool could also happen via a companion smartwatch app.

Gaming - The commercial success of titles like Nintendo Wii Sports (the fourth best-selling video game of all time [304]) and XBox's Dance Central franchise ushered in a new era of more physically-oriented games. While some mobile games have taken advantage of inertial or spatial data, none have leveraged arm pose to date. In particular, this could be an interesting opportunity to make smartwatch games larger and more interesting than their diminutive screens might otherwise allow. For instance, users could practice their golf swings or frisbee throws, with the watch reporting on achieved range and form quality. While I only considered arm pose tracking with the phone stored in the pocket, my technique should also extend to a phone held in the hand (with a different machine learning model). This would allow for gaming experiences with coordinated graphics. More specifically, the user would look at a game running on the phone held in one hand, with the other hand free to perform game actions, such as throwing a ball or fighting with a sword.

Context-Aware Interfaces & Assistants - Devices that are context-aware can automatically summon relevant interfaces and information to augment a user's activity. Arm pose is a useful piece of information that could improve the accuracy and breadth of prior context sensing work (using e.g., environmental sounds [150], bioacoustics [153], movement data [151]). For instance, arm pose tracking could help disambiguate if a user is driving an automobile vs. being a passenger. As discussed in the above subsections on Life Logging and Occupational Safety and Training, there is a long tail of arm-centric activities that could be automatically detected, which could launch context-aware functionality.

5.7 Limitations

I believe my proof-of-concept implementation demonstrates the feasibility of my method, but there are nevertheless limitations of note. First among these limitations is that my current work only considers the case when the phone is stored in a front pocket, a common but not ubiquitous placement [120, 234]. In general, my pipeline should scale to other pocket locations, including rear pants pockets, and potentially even cycling jersey pockets and fanny packs. However, when the phone is taken out and held in the hand, and can exist in many possible positions with respect to the body, a third spatial reference would be required (potentially head position captured via the phone's user-facing camera or earbuds with UWB). There already exists work that can automatically detect pocket vs. hand phone placement [98, 303], which could toggle between different models.

Another piece of information I need is which pocket a user typically stores their phone and which wrist the user wears a watch, which could be captured during a one-time setup wizard. Even with all this data, the user still needs to perform a calibration procedure, which is a limitation of my current implementation. This process is described in the Evaluation section, but briefly here, participants in my study had to hold the devices near each other, then assume a T-pose with their arm to orient the IMU reference frames (e.g. to offset how the phone is placed rotationally in the pocket). While inconvenient to perform once, I found the sys-

tem was able to maintain its calibration for an extended period of time. I note the T-pose used is not special and could be replaced by a pose users more naturally perform, such as arm by side (potentially auto-detected using UWB distance). If the calibration process could be condensed to a single gesture, such as holding the arms by the side of the body, I imagine there are many contexts in which it would be acceptable to do this once at the beginning of an activity, such as starting a workout or a rehab session. This is one aspect where the enhanced tracking mode for the iOS UWB API mentioned in the Implementation section could be particularly useful. This mode provides both distance and 3D angles between the phone and watch. Thus, if the watch was held in the phone's FoV cone, the phone would know its absolute position relative to the watch and could use this to use this data to calibrate the IMU reference frames automatically.

One additional issue is that many people do not carry their phone in their pocket at all [120, 234]. Feminine clothing lacking functional pockets [266] is a prominent example. People may opt to carry their phone in a bag or purse. Tracking watch position with a phone in a bag is a more challenging problem than a phone in a pocket because, while there is variation in pocket design, the phone is located in the same area on the body in the majority of cases. This is not true of bags where the space of possible phone locations is much larger. Since the phone could move inside bags, I do not expect SmartPoser to work well in such situations. However, as a solution, instead of using a phone, my system could generalize to other devices that are worn in static locations and may become UWB-enabled in the future, such as smart glasses or earbuds.

5.8 Summary

In this chapter I presented SmartPoser, a system that enables 3D arm pose tracking using an off-the-shelf smartphone and smartwatch in their normally-worn positions. In my approach, I build upon a state-of-the-art bidirectional LSTM model for arm pose prediction and fused more traditional inertial data with new UWB time-of-flight range measurements between the two devices. Independent of using commodity devices, this technique is novel as UWB ranging has never been used to augment IMU data in a wearable pose tracking system before. Through a user study involving real-world activities and gross arm motions, my approach yielded a median positional error of 11.0 cm for the wrist and elbow joints, with no user data in the training set. This is a level of tracking fidelity that has only been achieved in the past through full-body instrumentation with IMUs or UWB-instrumented environments. Overall, by achieving this result in this consumer form factor, I believe SmartPoser provides a realistic pathway for arm-based tracking and interactions to be adopted for widespread use.

In the next chapter, I will take a different approach to the one I used in SmartPoser and instead create a wearable utilizing entirely bespoke sensors. This device senses not only the arm pose of the user, but also can generate a 3D representation of their hand and objects currently held in the hand.

Chapter 6

Multi-View Depth Sensing From the Wrist

This chapter was adapted from my published paper:

> Nathan DeVrio, Chris Harrison. 2022. "DiscoBand: Multiview Depth-Sensing Smartwatch Strap for Hand, Body, and Environment Tracking" *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '22).*

6.1 Introduction

Hands are the chief appendage with which humans manipulate the world around them, and for this reason, digitization of the hands for use in interactive computing systems has been sought after for half a century [54, 354]. Applications vary tremendously, including smart environments with hand tracking [33], sign language recognition [268], gesture sensing smartwatches [55, 80, 111, 134, 345], and whole-hand replication in virtual reality for object manipulation [140].

Approaches generally fall into one of three categories. First, I can instrument both the user and the environment, for example with optical [287], acoustic [228], magnetic [225, 231] or other markers worn by the user and sensed by external

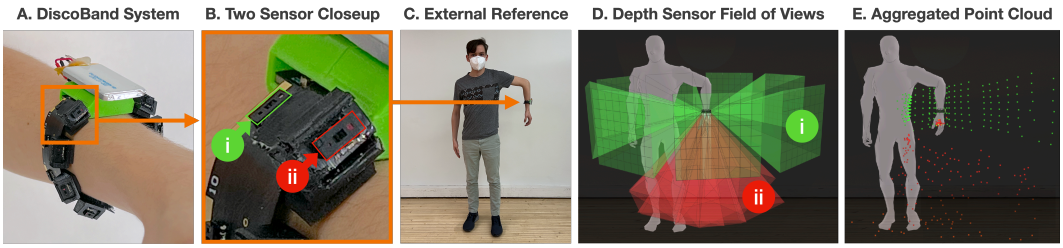


Figure 6.1: DiscoBand is a smartwatch strap (A-C) featuring sixteen ultra-small, multi-zone depth sensors (closeup of two sensors in B) looking outwards and towards the hands (green and red frustums in D), which capture a fisheye point cloud (E). Many interactive uses are possible, including arm and hand pose tracking.

sensors. Second, it is possible to instrument only the environment, for example with cameras and use computer vision to track a user's body pose [41]. Finally, I can instrument only the user, for example with body-worn IMUs [322], cameras [262], and many other types of active and passive devices (which I review later). The latter category has the significant benefit of being mobile (i.e., self-contained and not limited to a specially-instrumented room), and generally encounters less occlusion from objects in the environment (e.g., furniture) and the user themselves, depending on the instrumentation point.

The wrist is a particularly popular instrumentation point from which to sense the hands for three key reasons. First and foremost, it is a common place to wear jewelry, watches, and other bands. Second, it is a highly practical location to affix a small device to the body [78]. And third, wrists are proximate to a user's hands, offering the potential for superior data capture. For these same reasons, I too focus on the wrist location. Also similar to prior work, I employ optical sensors to capture the hand pose. However, as I will discuss, many prior systems employing such sensors have had to elevate components centimeters above the skin in order to achieve reliable line-of-sight, which results in form factors less amenable for consumer adoption. Additionally, techniques using cameras tend to elevate privacy concerns (unless otherwise noted, when I discuss "cameras" in regards to related work, I am referring to commonly-used, high-resolution, RGB or infrared cameras).

In response, I set out to create a sensor band that is comparable to a smartwatch strap: thin and self contained. For this, I use low-resolution (8×8 pixels), ultra-small (4.9×2.5×1.6 mm) depth cameras. To mitigate natural occlusion (e.g., fingers blocking line-of-sight to other features), I use eight sensors distributed around the periphery of the wrist (Figure 6.1, B & D, red frustums). When one view is occluded, the others are generally not, and in this manner they can work together to composite a live 3D point cloud (Figure 6.1E, red points) and resolve a probable hand pose.

My band also features eight additional depth sensors facing outwards (i.e., normal to the skin; Figure 6.1B), which I use to track the arm and upper body (Figures 1, D & E, green frustums and points). Taken together, these sixteen depth cameras capture a fisheye-like 1024 point cloud (Figure 6.1E), the rays of which are reminiscent of a disco ball reflecting light, and so I dubbed my prototype DiscoBand (Figure 6.1). The sensors I use have a range of 4 m, allowing me to also capture the proximate environment, opening other application areas discussed later.

Overall, DiscoBand offers a unique combination of features and properties that differentiate it from prior work. First and foremost, my band is thin, and could be plausibly integrated into future smartwatches. Second, my multi-view approach is inherently more robust to occlusion than single-view methods. Third, my low-resolution depth data is more privacy preserving than conventional camera-based wrist systems. Finally, my band's unique design and data opens entirely new capabilities not previously demonstrated with wrist-worn setups, including the ability to estimate user upper body pose, detect held objects, and scan the environment for obstacles and contextual clues. In this paper, I document the implementation of DiscoBand and different example applications I explored, as well as report results from a series of user studies that underscore the potential of my approach.

Reference systems	Tracking output	Method	High information	Low calibration	Low occlusion	Privacy preserving	Low profile	Low power
Myo [29, 32, 39]	Continuous pose	Electromyography (EMG)	✗	✗	✓	✓	✓	✓
Jung et al. [28]	Discrete gestures	Pressure	✗	✗	✓	✓	✓	✓
SensIR [41]	Discrete gestures	IR tomography	✗	✗	✓	✓	✓	✓
Interferi [27], EchoFlex [42]	Discrete gestures, cont. finger angle	Ultrasound tomography	~	✗	✓	✓	~	✗
Tomo [75]	Discrete gestures	Electrical imp. tomo. (EIT)	✗	✗	✓	✓	✓	✓
GestureWrist [49], CapBand [63]	Discrete gestures	Capacitive	✗	✗	✓	✓	✓	✓
Sugiura et al. [61]	Discrete gestures	Skin deformation	✗	✗	✓	✓	✓	✓
Serendipity [67], Xu et al. [71]	Discrete gestures	Motion/inertial (IMUs)	✗	✓	✓	✓	✓	✓
Beamband [26]	Discrete gestures	Ultrasonic beamforming	✗	✗	✗	✓	✗	✗
WristWhirl [17], RotoWrist [51], ThumbTrak [62]	Discrete gestures, cont. wrist angle	IR rangefinding	✗	~	✓	✓	✓	✓
Arakawa et al. [1], Chen et al. [8], Back-Hand-Pose [68]	Continuous pose	High-res RGB camera	✓	✓	✗	✗	✗	✗
Digits [30], Opisthenar [73]	Continuous pose	High-res IR camera	✓	✓	✗	✗	✗	✓
WatchSense [56]	Cont. pose (of second hand)	High-res depth camera	✓	✓	✗	✗	✗	✗
FingerTrak [25]	Continuous pose	Low-res thermal camera	✗	✗	✓	✓	✓	✗
DiscoBand	Continuous pose	Low-res depth camera	✗	✗	✓	✓	✓	✓*

Figure 6.2: A high-level overview of wrist-worn, hand tracking methods. I include reference system(s) for each method, which I then evaluate across a range of desirable qualities. Orange tildes are used when the capability is system-dependent (and less innate to the method). *Capable of low-power operation using the next generation of sensors [270].

6.2 Related Work

I now review three key areas of related work. First, I describe the many disparate application domains that can benefit from hand and arm pose tracking. I then move to a brief survey of hand and arm tracking systems worn on the wrist, but which sense the hands and body using indirect means. I conclude with a discussion of systems closest to my own: wrist-borne systems that use sensors to directly measure the physical configuration of the hands and arms.

6.2.1 Applications of Hand & Arm Pose Tracking

Gesture and pose tracking of both the hands and arms have been long standing goals in the field of Human-Computer Interaction (HCI). At a basic level, gesture and pose tracking can be used to augment existing devices, such as smartphones, with alternate input channels for interaction. Examples of this include zooming by pinching the index finger and thumb together [323], dismissing notifications with a flick of the wrist [30], and sign language input [268]. Recognition of specific key poses is particularly important in domains such as eating monitoring and medication adherence in healthcare [62, 235, 348], form correction in fitness [251, 276], avatar representation in virtual reality [49,244], and remote control of robots [291]. Whole-arm pose tracking has also been explored for in-air gestures for handwriting [334] and mapping symbolic body language to emojis [144].

6.2.2 Wrist-Borne Indirect Sensing

One approach to gesture tracking is through indirect sensing of measurable features that indicate the presence of a pose/gesture without ever imaging the shape of the body itself. Among the many methods in the literature, I was most interested in those that involved sensors worn on the arm or wrist, as this was most similar to my setup.

One of the most popular methods for indirect hand gesture sensing is electromyography (EMG), which measures the electrical activity of muscle tissue. Many

systems have implemented this technique because it is relatively non-invasive [137, 148, 173]. Similar techniques that also measure muscle contraction include using air pressure bladders affixed to the forearm [134] and resistive strain sensors on the wrist or back of hand [55, 165].

Another common technique for indirect hand sensing is tomography, which tracks pose configuration by emitting excitation signals into the forearm and measuring how the received signal changes based on shifts in the internal composition of the arm. There are many different ways to perform tomography, some involving infrared light [188, 217] or ultrasound [125, 189], while others use pair-wise electrical impedance measurements from electrodes surrounding the arm [345]. Less common indirect hand-sensing techniques include acoustic measurements (both active [205,264] and passive methods [95]), capacitive sensing [236,284], and measuring skin deformations on the back of the hand [278].

The last major indirect technique – detecting movement with inertial measurement units (IMUs) – is used for both hand tracking [302, 326] and whole-arm tracking [32, 67, 172, 254]. Advantages of IMUs include their ability to detect gestures beyond just the hand and their ubiquity in many of the smart devices users already own. It is worth mentioning that historically IMUs have been the only major technique that has been used for tracking arm pose using wrist-borne sensors.

6.2.3 Wrist-Borne Direct Sensing

Most similar to the sensing technique that DiscoBand employs are on-wrist systems that directly image the physical form of the hand or arm they are attempting to track. Within this category there are many different imaging sensors used. In past years, the two most popular methods have been 1) arrays of IR emitters/detectors positioned around the wrist that capture a single depth value per detector [80,81] and 2) single cameras mounted above the wrist of many varieties, including RGB [14,46,313], depth [267], thermal [327], and infrared [140,296,330]. Although not technically a wrist-borne system, ThumbTrak uses IR rangefinders in a ring array to capture finger micro-gestures [279]. Another prominent category of sensing is ultrasonic, ranging from single-sensor measurements [166] to complex beamforming arrays [124].

While often having superior SNR due to direct tracking of the hand, a commonality of the above systems is the need to operate significantly above the surface of the arm to achieve sufficient line of sight to e.g., the fingers [14, 140, 267]. Even still, if the wrist bends away from the sensor, most of these systems will lose tracking due to occlusion [313, 330]. One of the differentiating features of DiscoBand is that it aims to mitigate this occlusion problem by capturing depth maps from many overlapping view points around the wrist. This multi-view approach is underexplored, with the exception of FingerTrak [111], a hand tracking system using four thermal cameras positioned around the wrist. FingerTrak's thermal cameras are low resolution (32×24 pixels), and thus, like DiscoBand, the system is more privacy preserving relative to those employing high-resolution cameras. Of course, the two systems are quite different in terms of number of sensors used (and thus views captured; 4 vs. 16) and the type of sensor data captured (temperature vs. distance).

Finally, the most related systems from a sensing standpoint are WristWhirl [80] and RotoWrist [245], which track continuous wrist angle using small infrared range finders (depth sensors). The latter sensors are very similar in style and operation to the sensors I use, but are not multi-zone. In other words, WristWhirl's twelve and RotoWrist's eight sensors provide exactly that many distance measurements, whereas DiscoBand's sixteen sensors provide 1024 points in a 3D volume. Moreover, the specific sensor I use has a larger field of view, allowing DiscoBand to capture an aggregated fisheye-like point cloud. These improvements allow me to consider new and interesting use cases, discussed later.

Beyond IMUs, there has been relatively little work on wrist-borne arm tracking. Notable among non-IMU methods is work by Hori et al. [109], which used a wrist-mounted 360° camera to estimate full body pose. With a resolution of 4992×2496, there are obvious privacy implications, and the apparatus itself is approximately 10 cm in height. Even still, there is significant self occlusion, which has been an issue for other worn, single-camera, pose tracking systems. DiscoBand contributes a new method for arm pose tracking which is comparatively practical and compact.

6.3 Implementation

DiscoBand is a combination of hardware and software contributions working together to enable a series of use cases, which I subsequently evaluate. In this section, I detail the implementation of these components.

6.3.1 Hardware

My early hardware prototypes consisted of a series of linked rigid PCBs (example prototypes shown in Figure 6.3, top row), but these were awkward to wear and increased the height from the skin, one of the design parameters I sought to minimize. My final design (Figures 1 and 6.4) features two flexible PCB "wings" that permit my prototype to be closely wrapped around a user's wrist, emulating a smartwatch strap. Figure 6.3 (bottom row) offers a mockup of a commercial implementation using a flexible PCB over-molded in silicone.

The most notable component I use is STMicroelectronics' VL53L5CX time-of-flight (ToF) ranging sensor (Figure 6.1B, green and red highlights). This diminutive surface-mount component is able to capture an 8×8 depth image at 15 FPS, up to a range of 4 m. It uses a 940 nm class 1 laser, which is eye safe. Eight of these sensors face outwards from my band (i.e., normal to the skin), where they can image the wearer's body and environment. Each sensor has a 45° horizontal field of view, so that together, the eight sensors provide a 360° point cloud (Figure 6.1D, green frustums). A second set of eight sensors faces towards the user's hand (i.e., parallel to the skin), sitting on flexible PCB tabs that bend upwards. These sensors have overlapping field of views (Figure 6.1D, red frustums), but because they operate from different vantage points, they are able to image parts of the hand that might otherwise be occluded in other sensor views.

All sixteen VL53L5CX sensors are controlled by an STM32L431 microcontroller, which sits on a small rigid PCB to which my two flexible PCBs attach. I

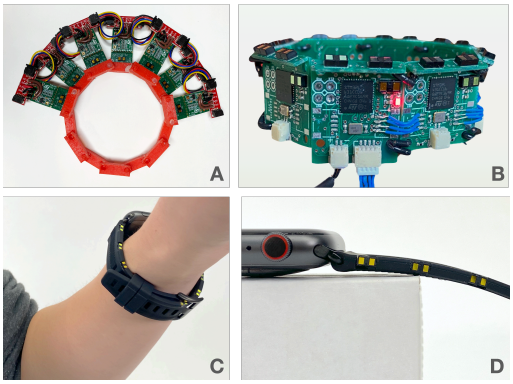


Figure 6.3: Two early DiscoBand prototypes made from rigid PCBs (A & B). A physical mockup of how DiscoBand could appear in a commercial form factor (C & D).

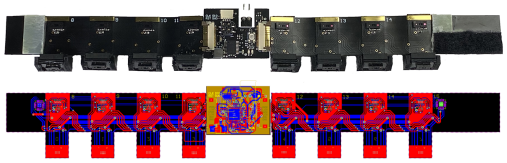


Figure 6.4: A photo and PCB layout of my final DiscoBand prototype, which can also be seen in Figures 1, 5-8, 13-16.

use all three of the chip's independent I2C buses to maximize sensor throughput. The rigid board also features a BNO055 9-DOF IMU (three axis inertial data, absolute orientation, and magnetometer), 64 kB FRAM, and analog power circuitry for the sensors. Using a modular serial port, my sensor band can run over USB (for power and data) or be battery powered and communicate over Bluetooth. I discuss power consumption in Section 6.8.

6.3.2 Firmware

My firmware handles three main responsibilities: interfacing with the depth sensors, packaging data frames, and transmitting them from the watch to a computer for processing. I note that in the future, all processing could occur on the watch itself, as the latest generation of smartwatches contain very capable processors and machine learning hardware accelerators. To maximize my framerate, I utilize all three I2C busses on the STM32L431 microcontroller, and also use direct memory access (DMA) to free the processor for other parallel tasks. One bus is devoted to each wing's array of hand-facing depth sensors, while the third bus communicates with all of the outward-facing sensors. I found that the VL53L5CX sensors do not significantly interfere with one another even when operating simultaneously, and I can further select sensor pairs that are on opposite sides of the body to further reduce any interference. This strategy allows me to overlap as many as three sensor read requests at once (two hand-facing sensors and one outward-facing sensor). Rather than wait for all sensors to be read before transmitting a single data frame, I send sensor values piecemeal as they become available, which helps to reduce latency. As I have half as many sensors on my two hand-facing sensor buses, I can receive all hand-facing depth maps at 7.3 FPS. For my single bus of eight outward-facing sensors, my band runs at 3.7 FPS, and this is also the frequency at which I send 9-DOF IMU data. This data is transmitted over serial via USB or Bluetooth. I note that the depth sensors I use can go up to 15 Hz, and that my prototype's slower framerate is primarily a consequence of using a single low-cost microcontroller and sending data back to a computer. If a superior pro-

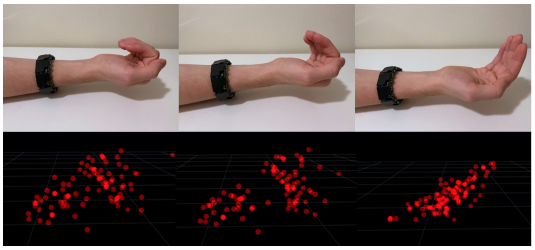


Figure 6.5: DiscoBand's depth sensors capture different views of the hand, which can be composited into a unified 3D point cloud. In this example sequence, the user uncurls their fingers, which is apparent in the point cloud data.

cessor was used, enabling on-device compute, both of these bottlenecks could be eliminated.

6.3.3 Compositing Multi-View Point Clouds

Although my flexible PCBs can deform, the relative geometry of the sixteen depth sensors is largely constrained. If I take into account a user's wrist circumference (which could be entered once during a setup wizard), I can make even tighter estimates. I use this geometry to composite all sensor data into a unified point cloud, as shown in Figure 6.5. I can orient this point cloud in three ways, useful for different applications. First, and most simple, is to keep the point cloud aligned to the wrist's coordinate system (essentially the band's first person view). Second, I can rotate it to align with the world using the BNO055's gravity and magnetic north vectors. Finally, I can align it to the wearer's body, using my upper body tracking described later. At this stage, I also perform basic signal filtering to reject outlier or intermittent noisy depth values, which improves the robustness of my downstream example applications.

At present, this multi-view compositing process occurs on a laptop using a custom application I wrote in Python. This app also offers a real-time visualization of the unified point cloud, with the ability to toggle on and off individual sensors, which was helpful for debugging and exploring possible use cases. However, I note that my compositing process is not computationally expensive, and could occur on the smartwatch in the future, as could the machine learning components described in the next section.

6.4 Example Uses & Implementations

My primary motivation for creating DiscoBand was to enable arm and hand pose tracking. My band's design reflects this, with two rings of depth sensors explicitly targeted for these applications. For these primary use cases, I built functional, real-time implementations, which I evaluate in a later user study. However, during my development process (and particularly after I was able to visualize the multi-view point clouds generated by my band), many additional uses came to light. These application areas, explored but not rigorously implemented, are described in Section 6.7.

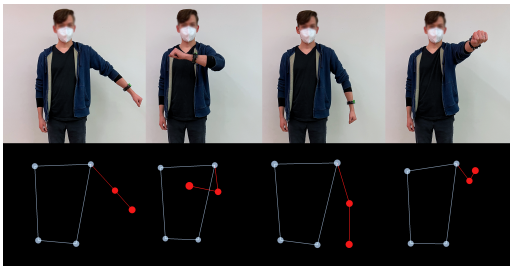


Figure 6.6: Four example arm poses (top row) alongside real-time output of DiscoBand's arm pose pipeline (bottom row).

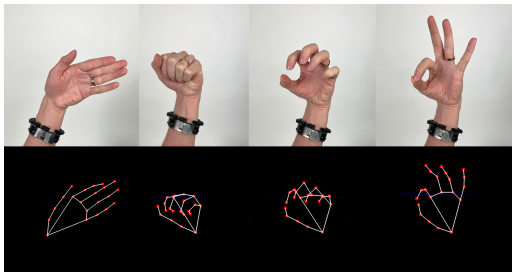


Figure 6.7: Four stills of example hand poses (top row) alongside renders of the 21 hand pose joints predicted by my model (bottom row).

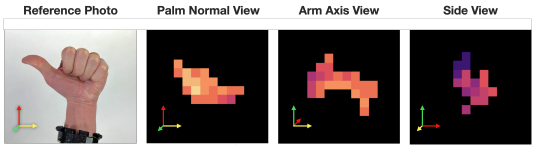


Figure 6.8: Left: photo of example hand pose. Right: the three virtual view depth maps my pipeline creates as features for hand pose estimation.

6.4.1 Arm Pose Tracking

Tracking a user's arm relative to their upper body has applications in context sensing (e.g., activity detection), fitness (e.g., rep counting), rehabilitation (e.g., range of motion tracking), and other domains. As discussed in Related Work, the most common way this data is captured today is with external sensors, such as cameras placed in a room. There are comparatively few self-contained, worn systems that track the arm and upper body in the literature, with the most popular method being worn IMUs requiring per-worn-session calibration.

For this application, DiscoBand uses data from its eight outward-facing depth sensors. The raw point cloud data is geometrically complex, and thus I sought an efficient way to reduce dimensionality. For this, I cluster the point cloud using DBSCAN [69] (min cluster size 16, max inter-point distance of 75 mm) and identify the closest large cluster (generally the user's torso). I then compute the phi, theta, and distance values as machine learning features. I also include the absolute orientation of the wrist (three Euler angles) provided by my band's IMU.

To capture ground truth arm pose for training, I use an RGB webcam and BlazePose [25] via MediaPipe Pose [83], which provides 3D estimates for 33 body keypoints. Of these, I save six keypoints: the left wrist, left elbow, left shoulder, right shoulder, left hip and right hip. In cases where a user wears a smartwatch on their right hand, I would capture right wrist and right elbow instead. I use SciPy's ExtraTreesRegressor (default parameters) to predict the aforementioned six upper body points, with the midpoint between the two hip joints as the root node. Example tracking is shown in Figure 6.6 and my Video Figure.

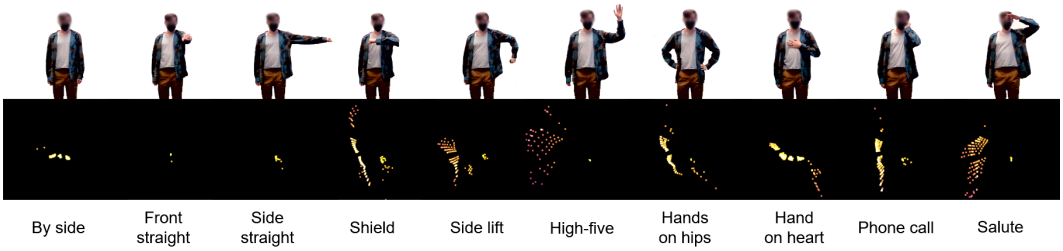


Figure 6.9: The ten terminal arm poses used in my study. Top row: Reference pose images. Bottom row: point cloud of body as seen by the eight outward-facing sensors.

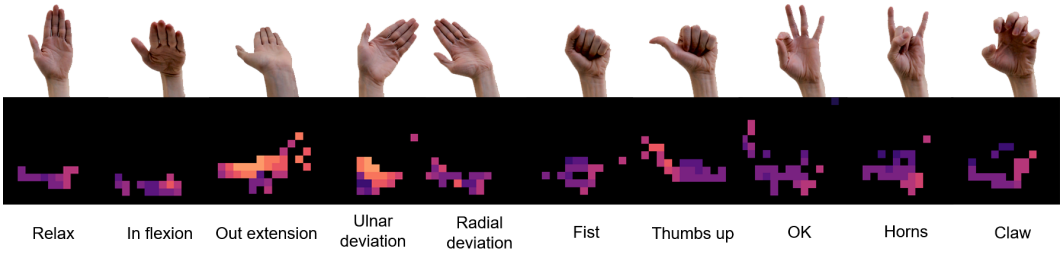


Figure 6.10: The ten terminal hand poses used in my study. Top row: Reference pose images. Bottom row: palm-normal-view synthetic depth maps **(see Figure 6.8)** of the hand as captured by the eight hand-facing sensors.

6.4.2 Hand Pose Tracking

More widely explored in the literature is hand pose tracking, which has immediate applications in VR input (e.g., grasping objects), free-space interactions (e.g., gestural control of IoT devices), and activity detection (e.g., carpal tunnel mitigation). I note that data capture inevitably suffers from heavy hand self-occlusion, and when rendered as a point cloud (examples in Figure 6.5), the data more resembles that of a 3D convex hull. Nonetheless, the external geometry that is captured is indicative of many different hand poses.

As with arm pose tracking, I featurize the point cloud to provide a lower-dimensional representation that is both descriptive and stable. More specifically, I create three synthetic 14×14 depth maps looking at the hand from different virtual viewpoints: looking down the palm normal, along the arm's axis, and a side view of the hand (pinky finger closest to virtual camera). An example of these three views for a thumbs-up pose can be seen in Figure 6.8.

For ground truth data capture, I use a webcam and MediaPipe Hands [82,174], which provides 3D coordinates for 21 hand keypoints. In pilot testing, I found this software offered better tracking accuracy and stability than a Leap Motion Controller, which is often used in such experiments. My model (SciPy ExtraTreesRegressor, default parameters) predicts MediaPipe's 21 hand keypoints. Example hand pose tracking can be seen in Figure 6.7.

6.5 Evaluation Procedure

To evaluate the efficacy of my multi-view depth sensing approach, I recruited ten participants (mean age 25, all right-handed) for a 60 minute study, which paid $20 in compensation. The study was conducted in a standard office space with large pieces of furniture nearby and next to a set of large windows letting in outside light. After a brief orientation, participants were fitted with DiscoBand, proceeding once they felt comfortable. I then recorded six body measurements: shoulder width, arm length, wrist diameter, palm width, band to wrist crease distance, and band to middle finger tip distance. These values are passed to my machine learning models as user descriptors, and also used to normalize my data in analysis.

My procedure was divided into two parts, starting with an arm pose study. Lacking a common set of arm poses from the literature to work with, I devised my own set of ten poses, exemplifying a variety of arm movements. In designing my arm pose set, I aimed to capture 1) a variety of joint movements (shoulder/elbow) and 2) joint orientations, as well as poses that 3) provided varied spatial endpoints (i.e., above/below the shoulders, as well as in-front/to-the-side of the body). Importantly, I was not just considering the poses as static endpoints, but also considering variation in the dynamic movements between poses. Pictures of my final pose set can be seen in Figure 6.9.

These poses were visually requested using a computer monitor. Participants were instructed to slowly move their body to match and then hold that pose for a few seconds, after which the monitor showed the next arm pose to be performed. One round of data collection consisted of all ten arm poses in a random order. Ten rounds of data were collected in this manner, which formed one session of data. DiscoBand was then removed and participants were given a few minutes break. The band was then re-worn and a second session of data was collected.

Importantly, I did not just record data at the terminal arm poses, but rather continuously throughout the experiment. This provided significantly more pose variety (and tracking challenge) than a defined pose set. Consider, for instance, all the intermediate pose states between the salute pose and hands-on-hips. In essence, each round of data capture can be thought of not as ten random pose trials, but rather as nine paired pose transitions (90 possible pairwise combinations). For ground truth 3D body pose, I used a webcam (1 m away) and BlazePose [25] running on MediaPipe Pose [83], which provides 33 body keypoints at 30 FPS. Each time my band transmits a frame of data, it is saved alongside the most recent BlazePose output. Across my ten participants, this data collection procedure yielded 50,000 body pose instances.

Next, my study moved to hand poses. This procedure was similar to arm pose, this time with a ten-gesture hand pose set drawn from the literature [124, 134, 345] (Figure 6.10). As before, a computer monitor requested the hand pose, and participants slowly moved their hands to match, holding the pose for a few seconds, before continuing to the next trial. One round of data collection consisted of all ten hand poses requested in a random order, and ten rounds were collected per participant to complete one session. As before, two sessions of data collection were completed, with a break in between when the band was removed. To capture ground truth hand pose, I placed a webcam 25 cm below the participant's hands, looking upwards. I use MediaPipe Hands [82, 174] to estimate the 3D hand pose

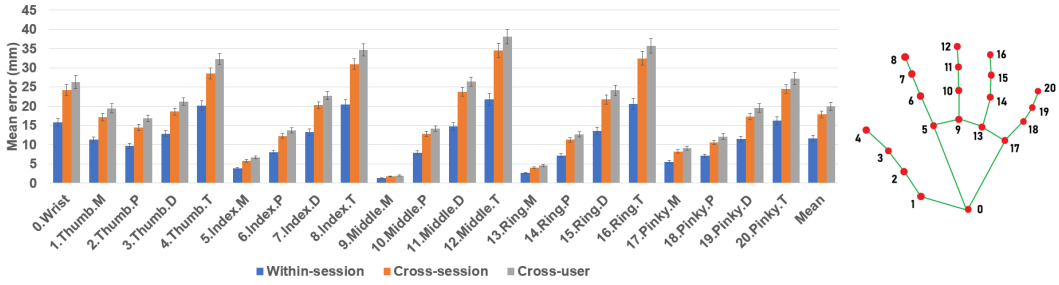


Figure 6.11: Hand pose study per-joint position error averaged across all participants for 21 hand keypoints. My lowest mean per-joint position error was 11.7 mm for within-session.

(21 keypoints) which, like before, is recorded in each frame of data. Across all ten participants, this collection procedure yielded 100,000 hand pose instances.

Similar to arm pose, I recorded the continuous motions between hand poses and not just the terminal poses. With this process, I was able to collect data and eventually track hand movements down to the granularity of individual fingers. Tracking of individual finger movements specifically occurred during the evaluation when performing transitions such as *Fist* → *Thumbs up* and *Relax* → *OK* and can also be seen in my Video Figure.

6.6 Results & Discussion

For my evaluation, I was interested in investigating DiscoBand's performance in predicting arm and hand pose when moving continuously between poses in my gesture sets. In addition, I also wished to quantify performance stability for a single user across multiple worn sessions, and across multiple users. To accomplish this, I compared my regression model results against the BlazePose ground truth in three separate analyses of within-session, cross-session, and cross-user performance.

6.6.1 Within-Session Performance

I conducted an investigation of within-user continuous joint error to simulate the performance of the system when it is calibrated to each user when first worn. I train/test my model using a modified k-fold cross validation procedure ($k = 5$) whereby I divide the data in the two session into 5 folds each, and then select one fold from each session to concatenate into a test data set and concatenate the remaining 8 folds into a training set. Of note is that all folds were divided at gesture transitions and data was not shuffled, ensuring there was never training and testing on data from the same instance of a performed gesture. I averaged the results from all 25 fold combinations to estimate within-session performance for a single user, and repeated this process for all users. Combining the per-user results, I then calculated mean per-joint position error (MPJPE) across all participants, which is shown along with standard error bars in Figure 6.12 for arm pose tracking and Figure 6.11 for hand pose tracking (dark blue bars).

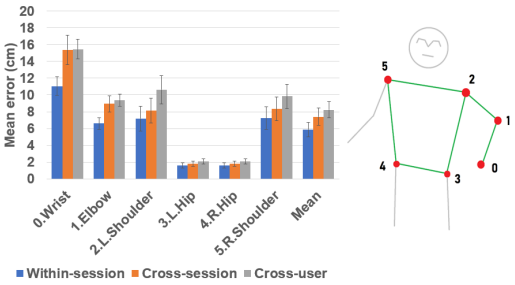


Figure 6.12: Arm pose study MPJPE across all participants for six upper body keypoints. Within-session accuracy was the strongest, with a MPJPE of 8.8 cm for the arm joints (keypoints 0 & 1) and 5.9 cm for all upper body points.

For within-session arm tracking, I found an MPJPE of 8.83 cm (SD=2.83 cm) for the arm joints (wrist and elbow) and 5.88 cm (SD=2.75 cm) for all upper body points. The maximum joint error was 11.05 cm for the wrist, which is expected given that it has the greatest range of motion and is furthest from the root node. When viewing arm pose prediction outputs, the most common observation was that the model would get the general arm gesture correct, and any error often arose from the fine pitch of the elbow or shoulder joint bends.

For within-session hand tracking, I found a MPJPE of 11.69 mm (SD=2.37 mm) and a maximum joint error of 21.84 mm (tip of the middle finger). Joint error decreases for each finger joint moving inwards from the distal end, as one would expect. Error was lowest at the middle finger metacarpal joint, rather than the wrist joint, because when the hand pitches MediaPipe returns keypoints rotated about a root close to that joint rather than the wrist, giving it the smallest range of motion.

6.6.2 Across-Session Performance

In addition to testing performance in single worn sessions, I also wished to test reproducibility, namely the ability to have stable performance each time the user puts on the band without the need for retraining. To test across-session performance, I evaluated my model with a simple leave-one-session-out cross validation, whereby I first train on data from session one and test on session two, and vice versa, averaging the results to obtain performance for a single user. This process was repeated for all users and all per-users results were combined to calculate MPJPE as shown in Figure 6.12 for arm pose tracking and Figure 6.11 for hand pose tracking (orange bars).

For cross-session arm tracking, I found a MPJPE of 12.16 cm (SD=4.32 cm) for the arm joints (wrist and elbow) and 7.40 cm (SD=3.28 cm) for all upper body points. For cross-session hand tracking, I found a MPJPE of 17.87 mm (SD=2.89 mm). In both studies, but especially for hand tracking, there was a noticeable increase in error when testing cross-session performance. This is almost certainly due to the band varying slightly in worn position across sessions, and given the model is only trained on data from one session (i.e., one location), it cannot extrapolate to other locations. It seems likely that more varied training data would

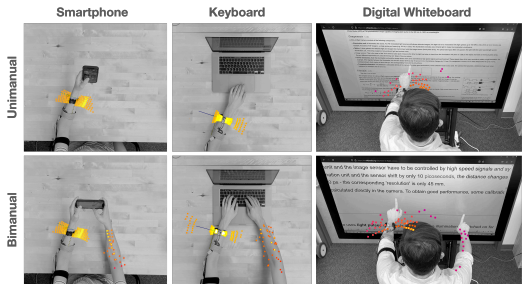


Figure 6.13: In these paired examples, I can see how DiscoBand is able to image the other (uninstrumented) arm. For illustration, the real point clouds have been rotated to match a greyscale reference photo.

be of great benefit. For arm pose tracking, the effect is less severe as surfaces are both farther away and more coarse. Finally, I note that cross-session robustness is often a challenge for worn sensor systems, and DiscoBand is no exception. As one point of reference, FingerTrak [111] saw a 127% increase in error when moving from within-session to cross-session testing (vs. DiscoBand's 53% increase in error).

6.6.3 Cross-User Performance

Beyond testing performance for individual users with custom models, I also wished to evaluate the ability of my system to be trained once (i.e., model flashed at the factory) and work for all users. To investigate this, I performed a leave-one-user-out cross validation, where data from nine participants was used for training and the tenth for testing. This process was repeated for all possible combinations of participants and the results were averaged to calculate MPJPE. The results are shown in Figure 6.12 for arm pose tracking and Figure 6.11 for hand pose tracking (grey bars).

For cross-user arm tracking, I found a MPJPE of 12.42 cm (SD=2.97) for the arm joints (wrist and elbow), and 8.24 cm (SD=2.96) for all upper body points. For cross-session hand tracking, I found a MPJPE of 19.98 mm (SD=3.29). I was surprised to find that for the wrist joint, the most important for arm pose tracking, there was almost no increase in error moving from cross-session to cross-user. This is in contrast to other approaches (e.g., EIT, tomography) which often see significant drops in performance across multiple users [125, 345].

6.7 Future Use Cases

As discussed in Section 6.4, my DiscoBand implementation focused on two core use cases: arm and hand pose tracking. However, over the course of more than a year of development, I identified several other interesting use cases. Although only brief explorations with basic implementations, I believe they convey how DiscoBand is an enabling technology with many interesting uses.



Figure 6.14: Example point clouds of various hand-held objects as captured by DiscoBand's eight hand-facing depth sensors. For illustration, the real point clouds have been rotated to match a greyscale reference photo.

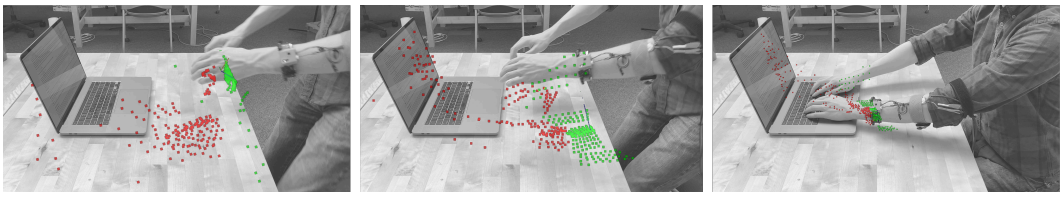


Figure 6.15: Example single-frame point clouds as the user approaches a laptop to begin work. Green and red point clouds are from outward- and hand-facing depth sensors, respectively. In the right two examples, the hand-facing sensors capture the geometry of the keyboard and screen of the laptop.

6.7.1 Bimanual Activities

While prior work has shown that smartwatches can be used to detect activities performed with the hand (see e.g., [150, 151, 153]), they are inherently limited by the fact that users wear one smartwatch on one arm. Worse still, the most common location to wear a smartwatch is on the non-dominant hand; if an activity is performed with the other (dominant) hand, it is likely impossible to detect.

In the case of bimanual activities — such as cutting food on a plate, steering a car, typing on a keyboard, riding a bicycle, tying one's shoes, or performing jumping jacks — DiscoBand can provide useful data. These examples are illustrated in Figure 6.13 where I can see DiscoBand readily captures the other arm using its outward-facing depth sensors. The resolution is crude at present, permitting only arm detection and angle estimation. However, forthcoming advances in sensor resolution will greatly enable this use case. I speculate that virtual IMU data could be synthesized by tracking the opposing arm mass, such that a single smartwatch has inertial data for both hands, which would be a huge boon to many context sensing applications.

6.7.2 Ad Hoc Touch Tracking

My band's hand-facing depth sensors not only capture the hand, but also objects and surfaces in front of the hand. These surfaces could be opportunistically appropriated for ad hoc touch input [97]. To explore this opportunity, I created a functional application that can detect touches and basic 2D gestures on ad hoc surfaces. To detect everyday planar surfaces (uninstrumented walls, tables, countertops, etc.), I run Optimal RANSAC [101] on the point cloud (ignoring points

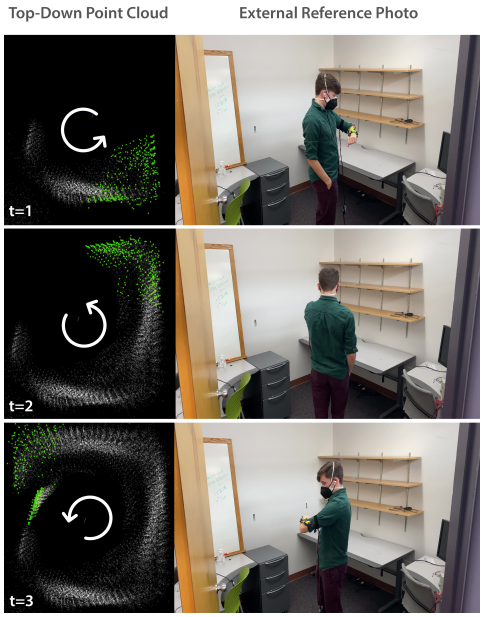


Figure 6.16: As the user turns, a higher-resolution, multi-frame point cloud of the office is built up. Note the ajar door to the left is captured.

further than 50 cm away). This process identifies and fits a plane to the point cloud, if one exists. For “click“ detection, I extend a ray from the band and test for collision with the fitted plane at the tip of the index finger (a distance I measure per user, but which could be captured once in a setup wizard). I note a more advanced version of my software would detect a pointing gesture using my hand pose pipeline, but I did not combine these two processes. Finally, once the user has ”clicked” a surface, my band tracks in-plane 2D gestures using its IMU (see Video Figure).

6.7.3 Held Objects

My eight hand-facing depth sensors are not only capable of imaging hand pose, but also objects held in the hand. Figure 6.14 includes example point clouds for a coffee cup, water bottle, notebook, duffle bag, grocery bag, umbrella and door handle. I threshold depth values further than 1 m, as these are unlikely to be an object held in the hand.

The low-resolution nature of my point clouds will naturally lead to confusion among objects if only geometry is leveraged for classification. However, machine learning could utilize two other important pieces of information. First is distance the object appears from the wrist, and the second is absolute hand orientation. For instance, an umbrella is not only distinctive for its large curved surface, but also the fact it operates 1 meter away and above the user. Arm pose data, which my system excels at, could also prove helpful in disambiguating some generic object geometries, such as a user bringing a cylinder-like object to their mouth (i.e., a bottled beverage).

6.7.4 Environments

While held-object and bimanual activity detection operate in the near field, I can also use DiscoBand to image the far field for environment detection and scanning. As noted previously, my current prototype can sense surfaces up to 4 m away.

Although a single frame of depth data does not typically allow for fine-grained geometric details to be resolved, some environments are distinctive. For example, a car interior is typified by a small enclosed volume, in which the hands operate in front of the body. Further, the presence of a vertical surface to the immediate left of a user would suggest they are in the driver seat (in countries with left-hand drive), which could be a useful contextual clue for a smart assistant (e.g., managing cognitive load). Figure 6.15 offers an example sequence of single-frame depth data, where the geometry of a tabletop and then laptop become visible. Even more interesting is when multiple point clouds are combined over time to build up a fine-grained 3D model of the environment. As a simple demo, I used my arm pose tracking (which provides 6-DOF wrist orientation data) to layer multiple point clouds. Figure 6.16 offers an example scene. Due to the nearly 360-degree nature of my sensing frustums, it is also possible to perform scans like this from a variety of arm poses, not just the one used in Figure 6.16. I note that more advanced multi-frame registration methods (e.g., SLAM [64]) would provide even better output, an implementation I leave to future work. Finally, I also speculate that a detailed model of the ground could be captured simply by the arms naturally swinging during locomotion, which could have assistive uses for those with impaired vision.

6.7.5 Body Scanning

Finally, DiscoBand is also capable of opportunistically scanning the user's body. The arms naturally sway in front and behind the user as they move about, providing multiple view points of the body. Like environment scanning, the point clouds can be correlated and overlaid to build up a 3D scan of the torso. Such a feature could be used to detect clothing, such as jackets, or potentially tracked over time for fitness and health applications.

6.8 Limitations

I believe my approach and proof-of-concept implementation demonstrates significant promise, but there are nonetheless important limitations worth discussing. Chief among these is the outstanding challenge of hand pose estimation in the face of significant hand self-occlusion. This issue is inherent to all wrist-borne direct sensing systems, and severity of the effect only increases as the sensors operate ever-closer to the skin. DiscoBand's multi-view approach partially mitigates this issue, and superior depth sensor resolution and machine learning models could yield further accuracy gains. Further, a multi-modal sensing approach, combining multi-view depth sensing with non-line-of-sight methods such as EIT [345] and EMG [137,148], could prove successful as they are robust in orthogonal ways [42].

Like almost all worn sensing systems, my band achieved its best hand pose accuracy when trained and tested using within-worn-session data (MPJPE of 11.69

mm). This result is not surprising, but it is also unrealistic, as consumers cannot be expected to re-calibrate their device each time it is worn. When I look at cross-session hand pose accuracy (i.e., where the band is removed and later re-worn), MPJPE increased to 17.87 mm, which is a more realistic appraisal of my band's pose accuracy. Even more challenging for the pose model is when it is only trained on data from other people, but never on its own user. This simulates "out-of-the-box" accuracy. Like most worn systems, across-user accuracy was DiscoBand's worst performing train/test condition. However, I was surprised to see that it was competitive with my cross-session performance. I suspect this is because the model was trained on more data (nine users instead of one), and hypothesize that across-user accuracy could improve further with a larger corpus.

My lab study also has some limitations, First, it was conducted exclusively indoors, as the one-off hardware was tethered and reasonably fragile. Although I did not formally test my system outdoors, it is expected that IR interference from direct sunlight would impact performance (though I note that time-of-flight depth sensors can operate in direct sunlight, unlike structured light approaches). Second, with regards to the biometric calibration data I collected from my participants at the beginning of the study, I would hope to eliminate this in the future. Instead of manual input of body data, it may be possible to conduct a one-time, automatic calibration where hand and arm length could be determined by performing known poses.

I also note that my wrist band is very much a proof-of-concept implementation, and several engineering challenges would have to be overcome in a consumer version. One issue is cost – my band cost roughly $200 to build (as a one-off prototype). While each VL53L5CX sensor only costs around $6 USD in large volumes, I utilize sixteen of these sensors that drives up the bill of materials. Perhaps in very large volumes, economies of scale could reduce the price. Alternatively, future iterations of the sensor could increase the field of view and sensor resolution, such that the number of sensors could be halved.

Power consumption is also a significant challenge, especially in a wrist-worn device where battery size is constrained. At full duty cycle, my present band consumes 3.6 W (which I note is lower than comparable systems such as FingerTrak [111]). For this reason, I completed much of my development and testing with the band tethered to a power source (runtime was 3.7 hours when using a 4 Ah LiPo battery). To make my approach compatible with mobile device power budgets, the sensing would have to be made intermittent, sampling the scene only occasionally. It could wake and run at full speed opportunistically, perhaps triggered by an application or IMU data (similar to raise-to-wake functionality in contemporary smartwatches). Fortunately, the time-of-flight depth sensors I employ are seeing continuous improvements. Indeed, the next generation of the sensors has already been announced by STMicroelectronics [269, 270] and will consume 50% less power than the components I used (while also offering higher resolution).

Finally, in this paper, I described seven example uses for DiscoBand: arm pose, hand pose, bimanual tracking, ad hoc touch tracking, held-object detection, environment scanning, and body scanning. I believe this long list highlights the generalizability of my approach and the long tail of applications it could enable. However, my explorations were not exhaustive, and I hope others will join me in

applying this general approach to new application domains, especially those that were not previously possible in a mobile context.

6.9 Summary

In this chapter I presented my work on DiscoBand, a novel sensing wristband utilizing a distributed array of ultra-small depth cameras to digitize a user's hand and upper body pose. My depth sensors also image the environment around a user, enabling additional uses cases. My approach allows for a uniquely thin form factor (<1 cm), which could permit integration into future worn consumer electronics. Further, my low-resolution point clouds are much more privacy preserving than comparable camera-based systems. In my user study, I found that DiscoBand can track the six upper body keypoints with a mean joint positional error of 5.88 cm. For hand pose, I found a mean joint positional error of 1.70 cm.

Through the work of DiscoBand, I showed how pose tracking can be enabled with depth sensors in a wristband that are low-profile and resistant to occlusion. I also discussed how pose can be an important implicit approach that can inform the nature of a particular interaction. In the final pillar, I am going to go one step further and show how I can make systems learn not only if your hand is assuming a particular shape, but whether you are using it to grab an object and engage in an activity.

I also want to highlight that so far, I have shown two practical methods for sensing implicit behaviors of users: where you are touching, and how your body is moving. Although I have shown these methods work well; they each are trained to track highly specific hand behaviors like when you make touch contact or your current pose. What these systems lack is a way to generalize to a broader scope of implicit behaviors like if you're brushing your teeth or driving a vehicle. In the next two chapters, I first discuss my technical approaches for detecting object grasps, then show how grasps can serve as a powerful trigger for egocentric camera systems that use vision-language models to recognize more general-purpose activities.

Chapter 7

Detecting Object Grasps With Sensor Fusion On the Wrist

This chapter was adapted from my paper:

> Nathan DeVrio, Roger Boldu, Eric Whitmire, Wolf Kienzle. 2025. "Contextra: Detecting Object Grasps With Low-Power Cameras and Sensor Fusion On the Wrist" *Proceedings of the ACM Conference on Mobile Human-Computer Interaction (MobileHCI '25).*

7.1 Introduction

Many people carry multiple wearable devices with them wherever they go. Yet despite their omnipresence, these devices still know little information about the settings users are in. As an example, a smartphone may be able to infer that a user is making dinner if they have opened a cooking app and are looking through a recipe, but very rarely is this done automatically. If obtained, contextual information, such as if a user is making a meal, can be used to provide better adaptive and context-specific interfaces and experiences. In this work, I explore how a device on the wrist is uniquely suited to sensing information that can be used to inform context.

Smartwatches are the most commonly worn all-day wearable devices; many users are already acclimated to wearing them. Smartwatches are also worn in close proximity to the hand, the primary body part people use to interact with the world. As a result, they are well-positioned to sense signals from interactions that are too noisy and too difficult to detect from other parts of the body. Past HCI research around wearable sensing on the wrist has actively explored recognizing general context by means of intermediate problems such as pose tracking and activity recognition. In contrast, there has been less exploration of directly recognizing specific moments of micro context, like picking up an object, when just-in-time feedback is most valuable.

In this project, I propose Contextra, a system that uses sensor fusion on the wrist to recognize when a user picks up an object. I reference Wimmer's and Zaiti et al.'s [286, 311, 341] claim that *object grasps* are a foundational component of

inferring when a user is in an interesting context that I might want to recognize and use it as the basis for this work's motivation. An "interesting" context relates to the idea that for some objects, if a user picks them up, it may be useful to the user to provide additional information or guidance to help them with their task. Examples include tracking caffeine consumed when picking up a coffee cup or providing explanation on how to use a tool. Wimmer's framework of grasp interactions [310] defines three stages of a grasp: capturing the grasp, identifying it, and interpreting it. In this work I focus on a technical, wearable solution for the first two stages. For the second stage, I focus on identifying whether a grasp occurred or not rather than recognizing the type of grasp from a known grasp signature. For the third stage, I discuss potential applications of Contextra in grasp interpretation, but leave this to future work.

Object grasps are complex actions, made up of multiple stages. Through experimentation I found certain types of sensors were better suited to detecting parts of a grasp than others. Past work has also shown the disadvantages of individual sensing modalities [178, 179, 187, 282]. Thus, using these factors as my motivation, I designed a system that uses a fusion of sensing approaches. On the wristband, I use optical, electromyography (EMG), and accelerometer sensors, each to detect different stages of the grasp. For example, the accelerometer is good at detecting initial movement of the arm and vibrations on object contact, while EMG can detect extension and flexion of the fingers when opening and closing the hand. Sensing the entire grasp process versus only trying to detect no grasp/grasp states has advantages for small objects and for EMG or acceleration whose features do not appear in steady-state signals. To clarify, data from the whole grasp process is used to pick up unique impulse features, but my target is to detect no grasp/grasp state transitions.

While EMG and accelerometers have been used in the past to try to detect grasps [29, 282], my incorporation of multi-view optical data from three low-resolution cameras is novel for such a wearable system. Beyond providing comparable grasp recognition performance on previously tested objects, I believe my camera fusion approach also has the potential to improve reliability and robustness across objects and users. This is because EMG and accelerometer data can be high variance depending on a particular grasp style [66, 192], while the presence of objects in the camera views tended to be a lower variance feature. These advantages show how fusing many sensors together is advantageous and how Contextra offers a unique contribution to the space of grasp detection that has not been filled by previous wearable devices.

I emphasize I tackle the problem of detecting grasps from continuous sensor streams, not just classifying the type of a known grasp. Compared to more common problems in prior work of classifying hand pose, grasp type, or what object is held, ours is a unique problem with its own challenges. I call this problem *continuous grasp detection* and believe it is an exciting problem because if solved, a system could be an automatic "wake word" recognizer for contextual AI, saving devices with conventional sensors like head-mounted cameras enormous power. In sum, in this paper I contribute the following:

1. A multi-view camera sensing approach on the wrist for detecting when object grasp events occur.

System	Worn position	Sensing method	Power consumption	Weight	Performance (within-session)	Performance (cross-session)	Performance (cross-user)	Objects
Pistohl et al.	Brain implant	Electrocorticography (ECoG)	n/a	n/a	0.89 recall	n/a	n/a	Drink mug
Xiao et al.	Wrist, forearm	Force myography (FMG)	~20 mW (*estimated*)	~40 g (*estimated*)	95% accuracy	n/a	n/a	12 x 6 cm cylinder
Maufroy et al.	Forearm	sEMG (Myo)	34 mW	93 g	96.8% accuracy	88.9% accuracy	82.8% accuracy	Hammer, 1 kg mechanical part, electric screwdriver, ratchet, plastic basket
Berlin et al.	Wrist, back of hand	RFID	210 mW	~35 g	72.8% detected (no ML)	72.8% detected (no ML)	72.8% detected (no ML)	Plastic bottle (L), plastic bottle (S), screwdriver, USB stick, hammer, box cutter, glass bottle, lipstick,
Theiss et al.	Wrist, back of hand	RFID, EMG, IMU	238 mW	~128 g	0.95 recall	n/a	n/a	Plastic bottle (L), plastic bottle (S), screwdriver, USB stick, hammer, box cutter, cardboard box
Cofer et al.	Wrist	Near-IR pair detectors	n/a	~60 g (*estimated*)	98% accuracy	n/a	n/a	Cylinder
DeVrio et al. (Contextra)	Wrist	IR cameras (3), EMG, IMU	255 mW	148 g	n/a	0.87 precision, 0.86 recall	0.82 precision, 0.84 recall	Plastic bottle (L), plastic bottle (S), screwdriver, USB stick, pencil, car keys, banana, notepad

Figure 7.1: High-level overview of highly-related worn grasp detection systems. Comparable numbers drawn from published materials as best possible, but please consult individual papers for specifics.

2. A late-fusion model that incorporates EMG and acceleration sensor features from the entire grasping process to overcome camera weaknesses like occlusion.

3. My entire system is low-power and privacy-sensitive so that it could run all day and prevent higher-power headset cameras and microphones needed for context-dependent apps from burning power and invading user privacy.

7.2 Related Work

In this section, I review past systems that used wearable sensors to look at objects held in a user's hand. I break out my discussion into three primary subsections. These range from (1) using held objects to recognize activities, the least similar to my task, to (2) classifying grasp types, and finally (3) detecting object grasps, the most similar to my task. Although there is overlap between my techniques and those used in grasp research for robotic hands, I omitted these works because of the gap in interaction task and the lack of human physiology to sense. The one exception is recent work on (4) sensor fusion to plan grasps for prosthetic robotic hands which I discuss in a final subsection.

7.2.1 Object-Centered Activity Recognition

The first category of work I examine involves systems that use only wearable sensors (i.e., no instrumentation of objects) to detect an object a user is holding and/or how it is being used to recognize a particular activity. These sorts of systems are unique from Contextra in that they do not sense grasp directly and are not focused on recognizing grasps, and instead focus on recognizing activities or use of held objects, which is different from my goal. Despite this, many of the sensing techniques used, particularly in camera-based systems are also applicable for my grasp recognition task and worth explaining because they both involve imaging objects near the hand.

Cameras are one of the most powerful sensors available for sensing object-based activities. This is because they are able to directly image if there is something being held in or near the hand. Maekawa et al. conducted investigations through multiple works showing how a camera on the wrist could effectively be used in combination with IMU and microphone data to capture information about what object a user is holding to classify activities of daily living [178, 179]. Independently, IMUs, particularly the accelerometer component have also been popular for recognizing object-based activities [37, 153, 204]. Common techniques when using IMUs involve sensing grasping posture and motion with held objects to get clues about the activity [37, 204]. Other less common sensing schemes that have been used to recognize object-based activities include magnetic sensors [176,177,297], capacitive sensors [243], and measuring vibrations from a voice coil actuator [212].

7.2.2 Grasp Type Classification

There is a separate body of work focused on the problem of classifying what type of grasp is being used. Knowing what type of grasp a user is using is often important because it can clue the identity of the object or what context or activity the object might be used in. Many different sensing approaches have been used on wearable devices to classify grasp types without instrumenting objects including EMG [6, 70], force myography [132], cameras [40, 140, 313], and inference directly from hand pose [352].

The approaches in these works are in general broken into two parts. First, sensing is used to detect what class of grasp is being used, such as a pinch, palmar, or cylindrical [6, 40, 70, 132, 313]. In some cases such as Fan et al's work using EMG, they also examined how performance varied for objects of different sizes and weights as these characteristics are directly correlated with the strength of EMG signals [70].

The second part of this category involves taking information about the grasp type and combining it with prior knowledge about a specific context or object set to identify what object the user is holding [40, 70]. In Section 7.6, I describe my vision for how Contextra could be combined with a camera on a headset to recognize objects in a power-efficient manner. Additionally, I discuss how objects could be directly identified from my camera data given prior information, similar to the works discussed in this section.

7.2.3 Object Grasp Detection

The last category of systems is focused on the same interaction problem as my system. Namely, given a continuous sequence of sensor data, can I identify at which moments, if at all, a user grasps objects? Systems in this category are most related to Contextra. As such, to show a direct comparison of Contextra to prior work, I provide a table in Figure 7.1 that provides a high-level overview of technical properties and performance for systems of this category.

Early on and to this day, the most popular approach for detecting grasps in this way has been using RFID. These systems instrument objects with passive RFID tags and place an RFID reader on the user's body to detect (via proximity) when

objects are picked up. Schmidt et al. [247] pioneered this technique with a more abstract goal focused around interactions with objects once held and used RFID reader prototypes integrated into clothing. Since then, many systems have iterated on the idea, compacting the prototype into a glove [75] or wristband [29, 74, 282].

While reliable, a major drawback with using RFID is that it requires all objects to be instrumented with tags ahead of time. To overcome this, many other sensing techniques have been used to detect grasps either instead of or in addition to RFID. The most popular of these involve sensing biosignals generated from muscles or neurons that are characteristic of the arm grasping an object. Example biosensing techniques that have been explored in the past include EMG [187, 282], force myography [321], and electrocorticography [224]. While biosignals can generate strong features and avoid object instrumentation, their practical use is limited by the high variance of their signals across different user populations.

One of the most promising ways to retain the benefits of biosignals like EMG while overcoming their limited robustness is to use data from additional sensing channels such as accelerometers [282]. Outside of EMG, tiered sensing has also shown promise when combining channels such as accelerometers and near-infrared detector pairs [50]. Two of the systems just mentioned are the closest to Contextra's approach, thus I describe them in more detail here.

Theiss et al. [282] created a wearable system for predicting grasps using a tiered approach to sensing to minimize power consumption. By default, an accelerometer was running, then when motion was detected, the EMG system was woken up. Then if a grasp was detected, RFID was used to recognize what object was picked up. Their grasp detector is notable because of its goal to detect grasps by using a fusion of features derived from accelerometer and EMG data. I note a critical difference that Theiss et al. used an EMG (Myo) band placed on the upper forearm while ours was worn at or close to the wrist. Sensed EMG signals are much stronger where muscle is thicker near the elbow, however this location is not practical for a system like ours that envisions itself as a part of a future smartwatch.

In Cofer et al.'s [50] work on the other hand, they used optical sensor data from 20 near infrared emitter-detector pairs. The NIR sensors were placed on a wristband and aimed inwards towards the wrist to measure tissue displacement and detect when objects were touched (picked up) by users. When a grasp was detected, an IMU was used to classify the grasp type. Note the NIR sensors only collect a single brightness value each, thus are much lower resolution than the cameras I use. Although the sensors used in Cofer et al.'s system were less like Contextra than Theiss et al.'s, their 1D-CNN model is more similar to ours than Theiss et al.'s decision-tree based model.

Compared to these past systems, I believe Contextra presents a novel innovation in fusing IMU and EMG data with feature-rich optical data from multi-view cameras. No prior system has attempted to use wearable camera data (single or multi-view) alongside EMG and IMU data for grasp detection. For the problem of binary grasp detection, Contextra is the first system to fuse data from different sensor types together. Prior systems that employed multiple types of sensors only used them in tiered approaches where one sensor detected an event that was used to trigger another [50,282]. Finally, to combine my sensing channels, I took advan-

tage of a multi-layer, late fusion deep learning model that stabilized performance across objects.

In practice, this means my system is more accurate at detecting when objects are picked up than prior work, and can be expected to work more reliably across a wider variety of different object shapes and sizes. This is the primary advantage of my technique over those compared against in Figure 7.1. While my grasp performance is superior to some other systems, for those with comparable performance Contextra demonstrated an ability to work across a wider set of objects. For example, Xiao et al. [321], Maufroy et al. [187], and Theiss et al. [282] relied heavily on an EMG-based technique, thus performance was only strong for larger and heavier objects which required more muscle contraction. I believe my improvements in grasp detection versatility pave the way for a general-purpose context-aware agent that can provide feedback to a user based on what they are holding in a way not possible with previous techniques.

7.2.4 Sensor Fusion for Planning Prosthetic Hand Grasps

Robots always know when they are initiating a grasp. Thus, instead of recognizing when grasps occur, work in the robotics field is typically centered around separate problems of determining the correct pose to execute the grasp and planning the motion [135]. However, with recent advances in prosthetics, there have been new avenues of robotics research focused on using wearable sensors to initiate and plan grasps for prosthetic hands. These works differ in that there is a human wearing the sensors who expresses an intent to grasp. The primary sensing modality used is EMG to allow users to use natural muscle contractions to control the hand. Secondary sensing modalities such as eye-tracking, IMUs, and cameras have also been used in sensor fusion which is highly relevant to my system.

Deshmukh et al. [56] and Zandigohar et al. [342] used a fusion of EMG and a head-mounted camera to plan grasps. In both they created a camera-EMG fusion model. Deshmukh et al. focused on classifying grasp types from EMG data [56] and Zandigohar et al. segmented continuous EMG data into four grasp motion phases [342]. On the other hand, Shi et al. used a model combining eye tracking and a camera from a headset and EMG and an IMU from the arm to control a prosthetic and plan the pose of grasps [258].

While many of the techniques used overlap, there are key differences between these works and Contextra. The first key difference between these works and ours is the difference in task. Namely, these works focused on planning grasp hand poses rather than using existing hand poses and objects to detect grasp moments. Second, my technique has the novel advantage of using multiple lower resolution cameras worn on the wrist instead of a single high resolution camera on the head. This change allowed Contextra to consume much less power while achieving strong performance.

7.3 Implementation

My work on Contextra involved developing both novel sensing hardware and the software to process sensor data. When combined, these components could auto-

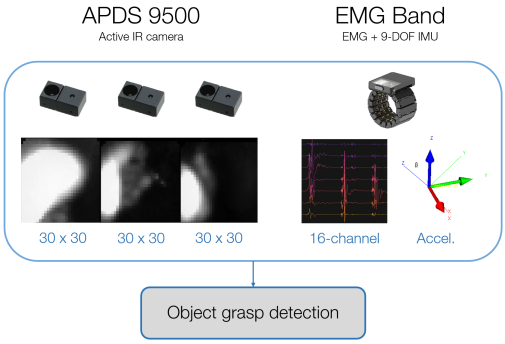


Figure 7.2: Contextra system diagram. I take 30 × 30 optical data from three mini active IR cameras and fuse it with 16 EMG channels and IMU acceleration. These sensing streams are combined in my model to perform object grasp detection.

matically detect when the wearer picked up objects. In this section I detail their technical background and implementation.

7.3.1 Sensor Selection

One of the most important questions in the initial design of my system was what sensing modalities to use. To decide, I broke down the anatomy of a grasp to identify opportunities for sensing. A typical grasp begins with the user reaching for an object by moving their arm. An IMU sensor, particularly an accelerometer, is suited for detecting gross motion such as this. Next, a user opens their hand (if not already), makes contact with the object, and closes their hand around it. An accelerometer may be useful again for detecting vibrations from contacting the object, but the extension and flexion of the fingers may be better sensed using EMG. In addition, cameras could be used to detect objects that have come into close contact with the hand and may be likely picked up. Finally, a user tightens their grip on the object and picks it up to complete the grasp. Depending on the weight of the object, EMG can be useful here to detect muscle strain during the lift. Otherwise, gross arm lift movement can be sensed using the IMU again.

While EMG and an accelerometer are powerful on their own, they do have certain disadvantages that make it hard to detect some grasps. A characteristic example is picking up objects slowly or lightly. A prominent EMG feature useful in grasp detection is the transient change in signal power. When picking up a light object, there is less strain put on the arm muscles so signal power changes less, to the point where in some cases it is barely noticeable from an EMG trace if an object is being held or not. Similarly, a prominent accelerometer feature is looking for signal spikes produced by vibrations when making contact with an object or surface. When objects are picked up quickly or aggressively, these spikes are prominent. On the other hand, when objects are picked up slowly or gently, the spikes can be nonexistent.

As a result of these disadvantages, I was motivated to pursue a sensor fusion approach where I combined data from multiple sensing modalities. Beyond EMG, and an accelerometer, which each have their own advantages for sensing different parts of the grasp, I searched for an additional sensing channel I could use to

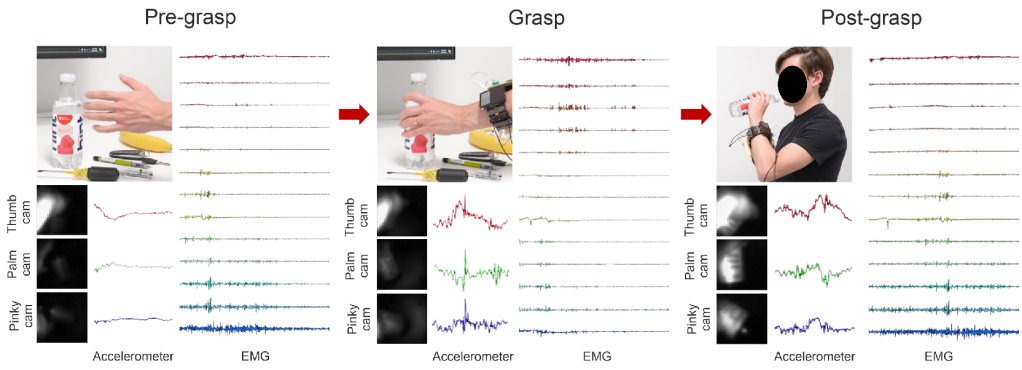


Figure 7.3: Timeline of a user grasping an object, along with demonstrative traces of sensor data from my different sources. In each frame, I highlight how different sensing channels (optical, EMG, and acceleration) can contribute by sensing different features of the grasp. For example, the motion of reaching for an object can be sensed by the accelerometer, then the object can be imaged by the cameras when opening the grip, and finally EMG can sense muscle contraction when lifting the object.

recognize objects held in the hand. In particular, I wanted a sensor that could use a static frame to detect grasps since transient features were already well covered by EMG and the accelerometer and could be diminished depending on the grasp. A promising candidate was to use optical data, such as from a depth sensor or camera, to see if an object was being held in that moment. However, optical sensors often have their own disadvantages, such as being vulnerable to occlusion, betraying user privacy, and consuming more power.

With all this in mind, I decided to take a balanced approach and used a series of three infrared (IR) cameras which were low-power and low-resolution and put together could prevent occlusion with multiple views. In more detail, the reason I used three cameras as opposed to one was because each camera can capture a different part of the hand and grasp that may be occluded in a single, static view. For example, a camera near the palm can see pinches of the index finger and thumb, but not objects that are approached from the side. This approach was inspired, in part, by other recent work taking advantage of multiple, smaller camera views for hand pose tracking [58, 111, 324]. I decided upon the number of cameras (not more or less) through an initial exploration in my simulator (detailed in subsection below) that showed three could well capture all visual aspects of the grasp without redundancies.

A diagram visualizing my chosen sensing channels can be seen in Figure 7.2. I note that despite my decision to use multiple sensors at the same time (mini IR cameras, EMG, and accelerometer), compared to a higher-res camera, total power consumption is still less and I still do not collect any data more privacy-compromising than 30 × 30 IR images (with background removed).

7.3.2 Camera Sensor Details

Contextra uses the APDS9500 optical sensors. These sensors are marketed to detect gestures using built-in blob detection algorithms. In my work, I used them

Component	Power consumption (mW) @ 3.3 V
Cameras (3 x APDS9500)	33 mW
EMG band (EMG + IMU)	150 mW
ML model (A15 Bionic)	72 mW
	Total = 255 mW

Table 7.1: Power consumption of each system component used for grasp detection.

as mini IR cameras by reprogramming the firmware to output raw camera frames over SPI. Notably, besides the imaging sensor, each component also includes an IR LED that can be used for active illumination of the target. Cameras were chosen as opposed to just photodiodes because of the number and quality of features cameras provide. Photodiodes may detect the presence of a target, but a camera can better see its shape to discern the target is a finger, not a held object.

When taking a picture of a target, I capture a sequence of three frames while toggling the LED from OFF-ON-OFF. Once complete, the two LED-OFF frames are averaged and subtracted from the LED-ON frame to produce, using in-situ hardware processing, a single 30 × 30 frame. Two LED-OFF frames are captured and averaged to reduce their higher expected noise relative to LED-ON frames. In the output frame, only the objects illuminated by the LED are visible (e.g., those close to the sensor), while all of the background is removed. An example output frame can be seen in Figure 7.4. The advantage of this approach is that the most prominent features in the output images are those which are salient to the grasp detection problem, namely the hand and objects near the hand. This makes the downstream recognition task easier by obviating the need to perform separate object segmentation on the camera frames to find the hand and objects. In addition, by removing the background, coupled with the fact that the cameras are already low-resolution, I betray significantly less privacy than higher-resolution cameras.

The goal of the camera sensors I used was to image objects held in the hand, thus one potential confounding factor is the presence of other, non-grasped objects nearby. While certainly a challenge, this is a situation that highlights the benefits of fusing other sensing modalities. For example, IMU and EMG would not trigger when just hovering near, but not grasping other objects. In addition, because of their limited field-of-view and proximity to the hand, many objects are naturally occluded and only those directly in or next to the hand are visible.

7.3.3 Power Consumption

Each camera outputs frames at 280 FPS. By reprogramming registers, I was able to lower the LED peak current to 56.7 mA and the exposure time the LED was on to 216 us. This led to only a 6% duty cycle for the LED and 3.4 mA average LED current draw which made each sensor only consume 11 mW during continuous operation. Considering that comparable small-package depth sensors and higher-res cameras can consume in the range of hundreds of mW [68, 271], using these sensors could result in an up to 10× savings in power. The low-power consumption of my optical sensing stack was one of the driving factors behind Contextra's overall design and purpose. Since it runs at such a low power, Contextra can run continuously to

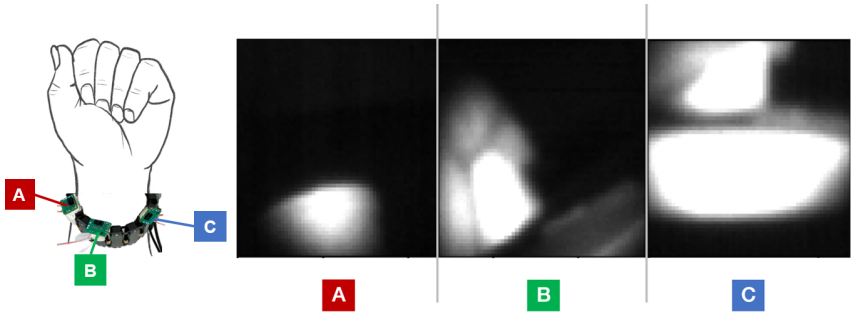


Figure 7.4: Example data from the three wrist cameras and mappings to their locations on the wrist; captured while picking up a paper cup from a table. These views show what kind of features can be picked up by each camera.

recognize grasps where other more powerful but higher-power systems cannot. In this way, even when Contextra cannot sense all parts of a context, it can provide a trigger to wake up other devices in a low-power sleep, such as smartglasses or a mixed reality headset, to perform deeper recognition.

Many existing commercial VR and XR headsets have built-in cameras used for pose tracking, however running and processing these cameras is a significant reason why they do not possess ”all day” battery life. The Meta Quest 3 and Apple Vision Pro, two popular headsets with such capabilities, only have battery lives of 2.2 and 2.0 hours, respectively [13, 193]. In addition, although such headsets already use cameras (thus camera-grasp detection would not increase power overhead), these devices are also typically used only for short sessions of entertainment or productivity. My target for Contextra is instead to augment devices like AR smart glasses that do not constantly run cameras and could have an ”all day” runtime. This is the reason I advocated for using less powerful but lower-power camera sensors.

Outside of the cameras, my band that collects EMG and IMU data consumes 150 mW and my estimate for the power consumption of my model on a mobile chipset is 72 mW (details in subsection 7.3.7). A breakdown of the power consumption of the entire system by component is shown in Table 7.1.

7.3.4 Comparison to Other Camera Sensors

When selecting the camera sensor, I sought a camera that possessed a small package size, low power, active illumination, high-frame rate, and a resolution large enough to sense object and hand features. The APDS9500 fit all of these criteria and thus I chose it because it was easy to procure and interface with. However, there are alternatives used by other systems with similar properties or minor tradeoffs.

Maekawa et al. used an unspecified camera which captures 352 x 288 pixel images at 6 Hz (no power specified) [179] OptiRing from Waghmare et al. used the OVM6948 and a red LED [293]. This sensor had a 200 x 200 pixel resolution, but was limited to 30 FPS and required 39 mW to run due to extra analog processing required of its signal. CyclopsRing by Chan et al. used a SONY CCD sensor that

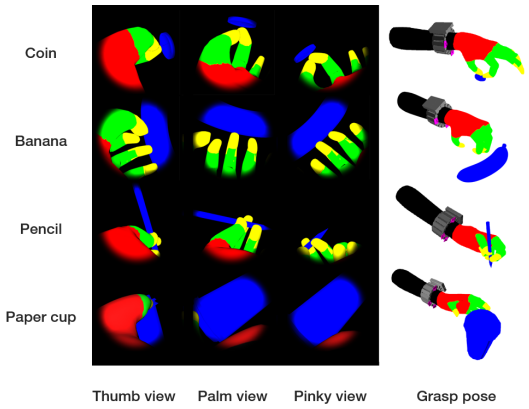


Figure 7.5: Views from the final camera positions of the thumb, palm, and pinky during a run of my simulator. On the right I show a rendering of the grasp pose looked like for each of the four test objects shown.

captured 640 x 480 images at 30 FPS [43] No power was listed but the similar OV7670 consumes 60 mW continuously [17]. General trends from these systems and other popular sensors (*OV2640*: 1600 x 1200 px, 15 FPS, 125 mW [16], *IMX708 Pi Camera 3*: 4608 × 2592 px, 10 FPS, >750 mW [15]), are that within this form factor, resolution can be increased, but this trades off frame rate and power consumed. These were two properties I wanted to optimize for, which justified my balanced choice of resolution, frame-rate, and power.

One of the most competitive alternatives is the Himax HM01B0. This sensor, used by Maruki et al. in their IRIS ring-worn camera, runs up to 60 FPS at 4 mW and is ∼2x as large as ours. There is no active illumination however; pulsing an external IR LED would result in a total power similar to ours and add size. Notably, the HM01B0 has a 320 x 320 pixel resolution, much higher than ours. While more resolution may help discern tiny objects better, many of the contextual features I detect (e.g., presence of hand and object) are detectable with a lower resolution.

While Contextra could operate with higher-res camera sensors, there are three main drawbacks to consider. First, high-res cameras (with some exceptions such as the HM01B0) tend to consume significantly more power [15, 16, 17]. Second, the higher the resolution, the greater the bandwidth burden, which can slow overall system speed. Finally, increasing resolution increases the risk to invade user privacy. Down-sampling a high-res camera could be used to alleviate this, but the other two drawbacks remain. Additionally, a lens with a short focal length could help to preserve privacy when designing a future high-res camera sensor, however using external lenses with existing sensors adds bulk.

7.3.5 Camera Position Optimization

In Contextra, I take three of the camera sensors and place them at different positions around the wrist. The positions, which are shown in Figure 7.4, include (A) beside the thumb, (B) in front of the palm and ring finger, and (C) beside the pinky finger. These positions were found, after experimenting with live data, to provide the most information on object grasps across all camera views.

I target the camera sensors to be placed on the front of a smartwatch band, in a normal worn position on the wrist. From experimenting with sensor data, I knew the rough positions I wanted the three sensors to be located on the wrist. However, to refine further and determine the exact position, pitch angle, and tilt angle of each sensor, I made a simulator that could take in 3D models of a grasp pose and an object and output the best sensor configurations.

The simulator modeled each sensor as a directional light source and camera with the correct field-of-view (45°) to match the real-world sensor. To ensure that the simulated cameras did not all converge on the same "best" view, I implemented strict boundaries on how far the sensors could move from their starting positions (thumb, palm, pinky), and how much they could tilt. This resulted in views which could image the same target, but retained the unique perspective from their respective wrist position.

With camera sensors, placing them too close to the hand can lead to occlusion by the palm [58]. This is part of the reason why my simulation was so important and I experimented with changing the sensor tilt angle. In my simulation, I assumed the smartwatch band was placed on the wrist, ∼3.5 cm from the base of the palm. During my Evaluation, I asked users to wear the real band in this same position, however due to a fixed minimum band size and differences in users' wrist sizes, the band was worn further up the forearm for some users (such as the model in Figure 3.3). Thus, although I did not formally evaluate band placement on the wrist/forearm, the wrist was the primary worn position and variations were represented in my data.

In the simulation all parts of the hand and object that could be imaged were painted a unique color according to their region of interest. The regions I painted included the palm (red), fingers (green), fingertips (yellow), and object (blue). The purpose of the simulator was to take input on whether it should reward having more or less of certain regions in each image and find sensor configurations that maximized this reward function. During operation, the following process was used:

1. Instantiate the real-world experimented sensor positions as origin points.
2. Define a range of possible sensor translations and rotations (from user input).
3. Iterate over all possible sensor configurations and capture a 2D image at each.
4. Use a weighted function to calculate a score for each image by looking at each pixel, assigning a score (0: black, 1: red, 2: green, 3: yellow, 4: blue) based on what color that pixel is, and summing all pixel scores.
5. Average the best angle and position configurations across different grasp poses to get the best overall configuration.

In my analysis, I picked a range of translations and rotations to match the physical constraints of the band. In my color-weighted function, I assigned highest weights to the fingertips and object (most important for grasp) and lowest to the palm (least important for grasp). After running this simulation on four different objects, while varying approach and wrist angles, I averaged the properties

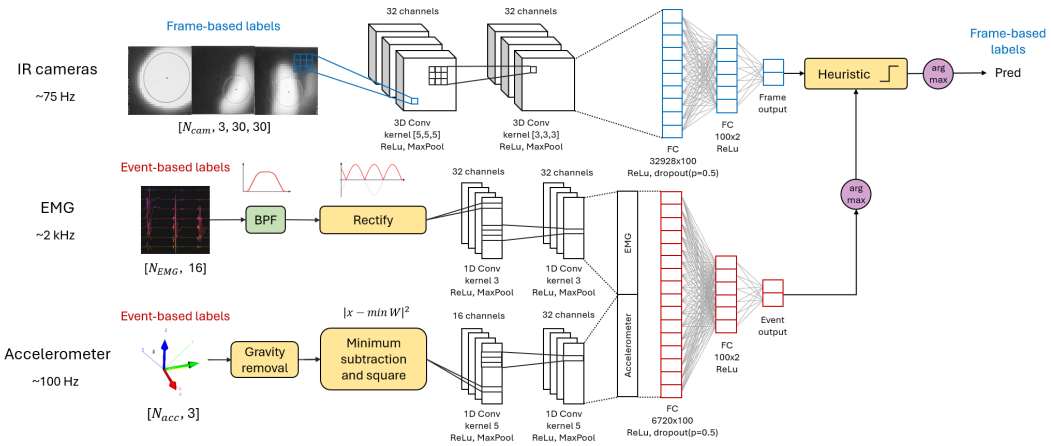


Figure 7.6: A diagram of my model architecture. My model was split into two branches based on ground truth label structure. For camera data, I used frame-based labels which indicated a touch or no touch state and for EMG and IMU data, I used event-based labels which indicated when grasp events (pick up or set down) occurred. EMG and IMU data were combined using a middle-fusion approach and then passed through a heuristic boost function to convert its output to match the frame-based labels. Once in a common format, the outputs from both branches were combined and returned with the frame-based labels.

of the best configurations. From here I made minor tweaks to account for outlier views, and produced a "simulated optimal" sensor configuration, shown for different objects in Figure 7.5.

The resultant configuration produced by my simulator is what I used to guide the mounting of my real-world sensors on the EMG band. Anecdotally, I noticed an immediate improvement in the features I was able to image in live camera data after switching to simulation-verified sensor positions. Outside of Contextra, I believe there is potential in this approach for other wrist-based optical sensing systems and believe it could be improved by using motion capture data to pose the 3D models.

7.3.6 Data Processing

A challenge when processing my data was that each sensor stream came from a different source, each with a different data rate, and with frame timing that could not be synchronized. This was a problem because my fusion model needs fixed-size CNN inputs. Without interpolation, if I received 26 EMG frames for 1 camera frame and 27 EMG frames for the next camera frame, my fused CNN input would not be fixed size. To manage this, I employed a buffered window approach. In this approach, data from all sources was buffered for one second. Each second, a timer was called that emptied these buffers and interpolated them to the expected FPS of each stream to account for possible dropped or extra frames. For example, for one camera, I expected 75 new frames each second. This was downsampled from the 280 FPS output of the sensor to improve model latency with negligible performance impact. If 74 or 76 were received instead, I interpolated *in time* to get exactly 75 frames. The effect of this on the final data was almost undetectable.

Afterwards, the one-second chunks collected could be rearranged and overlapped to produce fixed-sized, windowed inputs to the model. I used a 400 ms window with 20% overlap, which has been found to encapsulate the duration of a grasp [321].

The one second buffer I used in my implementation meant in live operation, predictions would take one second to arrive plus the inference time of the model (4.4 ms). I chose this buffer size to minimize interpolation needed to batch data across sensor streams. However, with a minor increase in interpolation, buffer size can be reduced down to 400 ms, the size of my model's window. This would cap the end-to-end latency of the system at $\sim$405 ms.

Before being passed into my grasp detection model, some of the data streams required extra processing. Aside from background subtraction and a normalization step the camera frames were not pre-processed and were passed directly into the model. On the other hand, EMG and accelerometer data benefited from small transformations. For EMG data, I applied a bandpass filter (cutoff frequencies of 20 Hz and 800 Hz) and performed full-wave rectification. The filtering helped remove baseline noise and motion artifacts while rectification improved interpretability of the signals.

Past EMG-based systems have found success with extracting features such as mean absolute value (MAV), RMS, waveform length, and covariance between channels and running these features through a random forest or SVM classifier [6, 70, 282]. I tried this approach but found better success by keeping the EMG signals as a time series and processing them with a 1D-CNN. This may be due to the fact that I was classifying EMG using event-based grasp labels that prioritized transient changes in the data.

For accelerometer data, I used a similar approach to EMG. First, I removed the gravity component from the accelerometer vector. I then took each window and subtracted the min of the window from the entire window and squared the result. The purpose of this transformation was to emphasize transient spikes in the data which is the main accelerometer feature I used for grasp detection.

7.3.7 Model Architecture

A diagram of my complete system architecture can be seen in Figure 7.6. This diagram includes the data processing steps described in the previous subsection and the machine learning models described in this one. Contextra's approach to recognizing grasps uses data fusion. In my model I decided to use two layers of fusion. First, the data from the cameras was separated from the EMG and accelerometer data. At their core, the data in these two groups looked at grasp from two different angles. The features from the camera data tried to answer the question "is there currently an object held in the hand?", for which a label is assigned to each frame to indicate if a grasp is occurring or not. On the other hand, for the time-series data, the features tried to answer the question "did a grasp just occur?", for which a label is assigned to an event in time. To simplify processing, I treated both the moment of picking up an object (lift off from rest) and setting down an object (touch down at rest) the same as "grasp events". As a result of this label distinction, one model with frame-based labels was used for camera data while a second model with event-based labels was used for EMG and accelerometer data.

Inside the frame-based model, a 3D-CNN was used. This was found to have better results than a 2D-CNN since it can capture temporal changes in image data, and was found to train more stably than a convolutional LSTM model. Inside the event-based model, separate 1D-CNNs were used for EMG and accelerometer to capture transient temporal changes in the signal traces. After this, the outputs of the 1D-CNNs were jointly fused before additional fully-connected layers. The output from both models were combined using a late-fusion algorithm.

In the algorithm, I use the raw output of my frame model as my base for prediction (before taking its argmax). I then look for transitions in the predicted output of the event model from 0 to 1 (I predict an event just started). When found, I add a boost B_e to the index of the frame output corresponding to the class predicted *least* in the last 5 frames. The purpose of this boost is to weight transitions in the frame model predictions when events are predicted by the event model. On top of this, a boost B_m is also given to the index of the frame output corresponding to the class predicted in the last frame. This second boost serves to give momentum to state predictions and prevent bouncing output. In my evaluation I used boost values of $B_e = 20$ and $B_m = 10$ which were found by experimentation on pilot data.

I used this heuristic because I found (A) the frame-based model was generally more reliable and (B) that transitions in the frame and event-based models were difficult to exactly match in time. As such it made sense to use the weaker event model to influence the transitions of the frame model but ultimately use the frame model for prediction outputs. In addition, the output frame-based predictions were converted to transitions again in my evaluation metric (see Results section) for a similar problem of aligning predicted transitions with ground truth transitions which using an event-based metric allowed me to overcome.

During testing, I found my model's performance improved if pre-trained on a larger corpus of pilot data. This pilot data was separate from the main training data collected on participants during the study. The size of this pilot data was roughly 4 hours of grasp events collected on data from one of the authors and two additional users (not in user evaluation). My model was pre-trained for 10 epochs on the pilot data then fine-tuned for 40 epochs with user data using the ADAM optimizer and a learning rate of 1e-4.

In total, the model has 613 M FLOPs and 4.07 M parameters. When tested on a Core i7-10750H laptop CPU the model achieved an inference time of 4.4 ms (227 FPS output). For reference, new frames arrive only every 13.3 ms which still gives a 8.9 ms buffer for real-time inference. For a mobile deployment I would expect my model to target a SoC such as the Apple Watch S9 which contains a 4-core neural engine. Exact specs for the S9 are not available, but its design is based on the A15 Bionic which has benchmarked an average latency of 9.44 ms (106 FPS) at 72 mW for a 67 M parameter CNN-based model 16x larger than ours [121]. Thus, I am encouraged my model could run at speed and remain low power on a mobile chipset.

7.3.8 Ground Truth Data

To collect ground truth data during my study, I used a Sensel Morph touch pad [250]. The pad detected when there was pressure applied to it, which I used as a



Figure 7.7: The set of eight objects used in my user study. Objects were chosen in part to parallel prior work and in part to offer a diversity of sizes, weights, and reflectances.

signal for if the object was currently set down. For the event-based model, I used transition points in the Sensel data (e.g., from pressure to no pressure) to signal when a grasp event occurred. In my study design, I considered a potential issue in that by detecting pressure, the Sensel would not detect the exact moment of the grasp but rather when the object was picked up and set down. Although an important distinction, this ended up not mattering with the technique for evaluating my data on the labels I adapted from Cofer et al. [50] and describe in Section 7.5. This is because a Sensel pick up transition always happens after a correctly predicted grasp up transition and vice versa for the set down transitions.

I also encountered a challenge when I discovered the outline of the pad (made of reflective metal) was clearly visible in the optical camera data. This worried me that my model may attempt to train on this feature instead of looking at the shape and outline of objects. I fixed the issue by laying a cloth over the pad and table underneath which provided a uniform and undistictive visual pattern.

7.4 Evaluation Procedure

To evaluate the performance of the system I developed, I conducted a user study on grasp detection accuracy. The goal through the study was to investigate the following research questions:

(RQ1) How well does Contextra recognize grasps of a diverse set of objects?

(RQ2) How much does Contextra's performance degrade if it is used by a user not included in the training data?

(RQ3) Are there certain types of objects that Contextra performs better or worse on than others?

My study was run on a group of 8 participants (8 male, 2 female, mean age 27). The study lasted approximately 45 minutes in length and was broken up into three identical sessions. Each session began with the participant putting on the Contextra device. I did not measure participant's hands explicitly, however anecdotally,

hand sizes varied widely (large to small) as well as large variations in wrist size and hair thickness on the back of the arm (which could be seen in camera images).

Participants then stood in front of a table on which was a Sensel touch pad and a collection of eight objects (shown in Figure 7.7). Half of the objects in this set were chosen to overlap with a common set from prior work [282]. The other half consisted of new objects designed to both provide more challenging targets and align with my envisioned example uses of Contextra.

During the session, a researcher placed an object on top of the Sensel pad, began recording data, and asked the participant to pick up and set down the object 15 times. I did not explicitly instruct users to perform challenging cases such as grasps with no object in the hand or resting the hand on a surface. Nevertheless, participants were asked to experiment with different typical grips they might use to pick up the object, and incorporate natural motions and actions while holding or not holding an object. For example: a participant is not holding an object and starts with their hand at their side. Upon being prompted to pick up a water bottle, they grab it around its base and hold it up to their mouth before setting it back down. Once complete, they return their hand to their side. The purpose of this was to capture grasp poses that users would use in the real-world and not just repetitive artificial grasps.

The goal of this instruction was to incorporate variety in the sensor data that would be more typical of how my system might be used in real-world interactions. While doing so makes the sensing task more challenging, it reflects the reality that users do not always grasp the same objects with the same grip or intensity. Further, a peak in EMG or accelerometer data on its own cannot reliably be assumed to be a grasp when working outside a lab setting.

Once 15 grasps were complete, this process was repeated for the remaining seven objects. At the end of the session, participants removed the Contextra band before re-wearing it for the next session. The purpose of this was both to give participants a break and simulate different worn positions of the band typical of normal use.

In my Evaluation, because of the nature of the ground truth sensor used, I only tested for cases where the object was picked up immediately after grasp. While sensor information from the entire grasp sequence (in a 400 ms window) was available to my model, the models were trained to detect ground truth transitions, in other words, the exact moment in time of the grasp. I did not explicitly test cases where objects were grasped but not picked up. These cases may be more challenging due to less IMU motion data, however the IMU acceleration and EMG spike features would still be present and could help maintain performance.

7.5 Results

In this section I detail the results from my evaluation. For each study, I also examine whether the research questions of interest I outlined in the previous section were met. To conclude the section, I provide a breakdown of performance with respect to different objects tested.

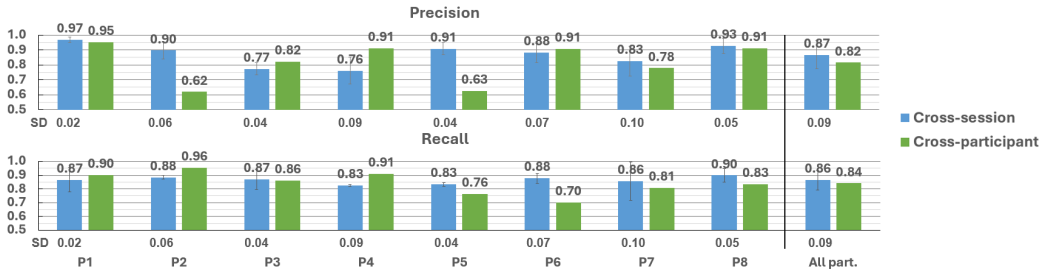


Figure 7.8: Average precision and recall were 0.87 (SD=0.09) and 0.86 (SD=0.07) for cross-session and 0.82 (SD=0.12) and 0.84 (SD=0.08) for cross-participant, respectively. Recall tended to be better and lower variance. P2 and P5 had drops in precision in the cross-participant case, potentially from unique physiology that highlights the need for larger and more diverse training data.

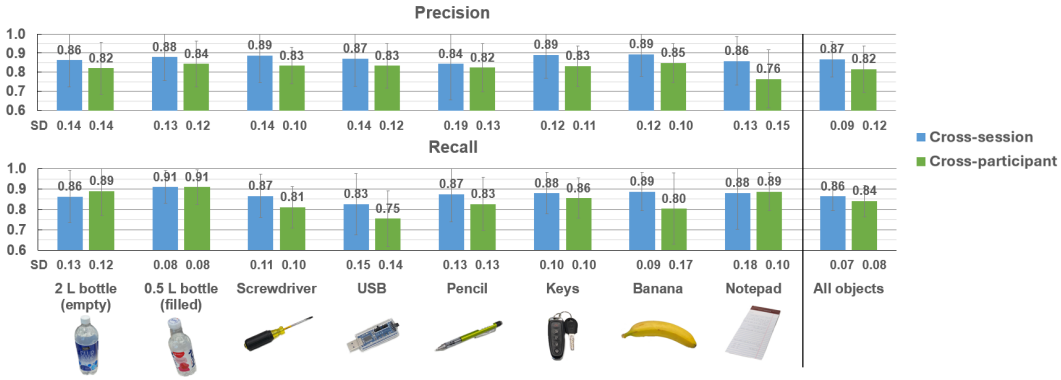


Figure 7.9: Precision and recall were fairly consistent across objects, with precision varying only 0.03 from the mean for cross-session and recall only 0.04. Performance was loosely correlated with the size of objects because small ones were harder to image. Small objects like the USB and pencil performed worst while large ones like the bottles and notepad performed best.

7.5.1 Calculation of System Performance

In my system, I calculate performance metrics for the percentage of correctly identified grasps, missed grasps, and spuriously detected grasps. I chose this approach because of the fact that my system continuously outputs predictions for a grasp state while my objective is to detect finite grasp events in time, thus measuring accuracy on a frame-by-frame basis is less useful. Before calculating metrics on my model's outputs, I first pass the predictions through a filter stage that looks for short oscillations between the low (no touch) to high (touch) states. If any state change is detected lasting less than 0.24 s, I determine it to be spurious and filter it out. This cleans up noisy oscillations in my output predictions. The chosen value (0.24 s, 3 window overlaps) was taken from Cofer et al.'s assumption of 0.16 s for minimal press length [50], where I added an extra window overlap of a window (0.08 s) after finding it increased stability.

When a change from the low to high state is detected in either label or prediction outputs, I check to see if the other time series also changed from low to high before the original series returns to the low state. If such a change is found, the event is correctly detected and marked a true positive. If not, the event is marked a false negative if it is initially detected in the label series, and a false positive if detected initially in the prediction series. This is the same technique used by Cofer et al. [50]. Improving their method, I also added extra logic to detect when low states are missed to ensure false negatives and false positives are correctly categorized.

7.5.2 Cross-Session Performance

To evaluate the results from my study, I performed two analyses. The first of these involved looking at performance across sessions. In this analysis, for each participant, I used a model trained on data from two out of the three sessions and tested on the third session. This process was then repeated for the other two sessions and results were averaged to get final results for each participant.

In my results, I tabulated the number of true positives (a grasp occurred and I predicted a grasp), false negatives (a grasp occurred and I did not predict it), and false positives (I predicted a grasp that did not occur). From these values, I calculated metrics for precision and recall to measure the predictive performance of my system. These results are shown in Figure 7.8. I chose to use these two metrics because of their widespread usage for comparability and their ability to show how well my system recognizes relevant grasp events. I also did not choose accuracy because we're detecting entire grasp events, not classifying individual frames, and accuracy would be diluted because there are many more no grasp states. Between precision and recall, I am most interested in high recall because my priority is to detect grasp events when they occur. Since we're trying to detect "wake gestures", recall measures how often the "wake" is not missed.

In my cross-session results, I calculated an average precision and recall over all users of 0.87 (SD=0.09) and 0.86 (SD=0.07), respectively. These results confirm RQ1. While recall was fairly consistent across participants, precision varied more widely with some participants having a much greater number of false positive triggers. I believe this behavior may be due to the volume of data I collected in my study. Curious, I tested using a smaller subset of session data and without

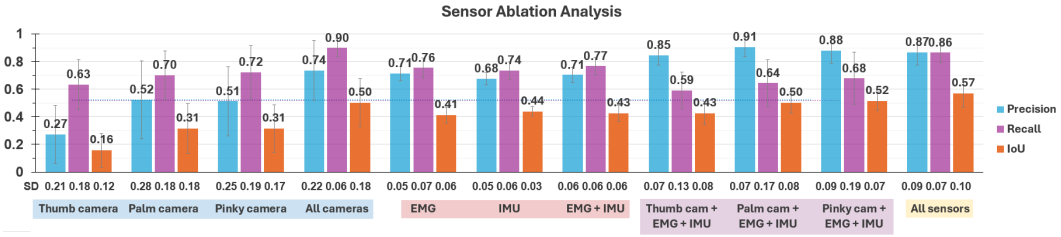


Figure 7.10: Ablation analysis for the different sensing channels of Contextra. Results are shown for mean precision, recall, and IoU from the cross-session participant data.

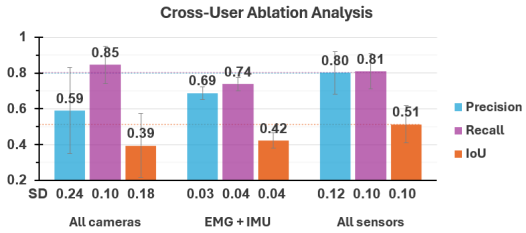


Figure 7.11: A subset of the ablation analysis specifically on cross-user data to highlight the advantage of adding cameras to EMG. Results are shown for mean precision, recall, and IoU.

pre-training and found that my model's precision improved greatly (>15%) for all participants when the size of the training corpus was increased. In the future, I believe this could be improved by collecting more training data of that user, and also pre-training with a larger corpus of data collected by many users. This is in line with past work showing the benefit of large training corpuses for EMG [223].

7.5.3 Cross-Participant Performance

In my second analysis, I used the same methods as the first one, however this time I was interested in examining how well the system performed when tested on new users it had never seen before. To do this, for a particular participant, I trained my model on data from all three sessions of all other participants and tested the model on all three sessions of the participant in question. The results from this analysis are also shown in Figure 7.8.

In my cross-user results, I calculated an average precision and recall over all users of 0.82 (SD=0.12) and 0.84 (SD=0.08), respectively. This result is lower than my system's performance for the cross-session case, highlighting the fact that due to the unique physiology of each user, the features they create are slightly different (answering RQ2). Thus, including data from the current user in training will improve the performance for that user. This result is also consistent with prior work using EMG sensor data which showed that although powerful, they tend to have a higher variance across users as compared to alternative sensing methods [66, 192].

7.5.4 Per-Object Performance

Next, in this analysis I break down performance for the different objects tested. In Figure 7.9 I show precision and recall for each of the eight objects. To answer RQ3, results were fairly consistent across all objects. For precision, all objects were within 0.03 of the mean for the cross-session case, with the lowest value for the pencil (0.84, SD=0.19) and highest for the banana (0.89, SD=0.12). For recall, all objects were within 0.04 of the mean for the cross-session case and variance was slightly higher with the USB performing the worst (0.83, SD=0.15) and the 0.5 L bottle the best (0.91, SD=0.08). These results are notable because as compared to prior work [29, 282], my system did not have a significant fall off in performance for smaller or lighter objects, indicating more robustness for recognizing grasps of diverse object types.

7.5.5 Sensor Ablation Analysis

In my final analysis I performed an ablation study on the contribution of each sensing channel to the results of my entire system. For this analysis I used my cross-session data and trained and ran separate models for all major configurations of sensors. Specifically this included breaking out results for: only one camera, all three cameras, only EMG, only IMU, EMG + IMU, EMG + IMU + one camera, and all sensors.

7.5.5.1 Precision and Recall Analysis

Mean results for precision and recall are shown in Figure 7.10. Overall, results were the worst when using one camera. Of the cameras, the thumb camera was the worst performing with a low recall (0.27, SD=0.21) while the pinky camera performed best, albeit with a still low recall (0.51, SD=0.25) but the highest precision (0.72, SD=0.19). This is logical given that the pinky camera has the best view of objects as they are approached while the thumb camera may only see them, if at all, once picked up. On the other hand, combining all cameras provided a significant boost to precision (+0.22) and recall (+0.18) versus the best single camera (pinky).

For the other sensors, the cases for just EMG, just IMU, and EMG + IMU all performed very similarly (both metrics within 0.03). Of these, EMG + IMU performed best for precision (0.71, SD=0.06) and recall (0.77, SD=0.06). Compared to the all cameras case, EMG + IMU had a slightly worse precision (-0.03), and a much worse recall (-0.13).

I also examined the cases of adding just a single camera to the EMG + IMU case. By doing so, precision significantly increased (+0.20 at best) while recall decreased (-0.18 at worst). The best camera sensor to combine with EMG + IMU was the pinky cam which had nearly the best precision (0.88 SD=0.09) and clearly the best recall (0.68, SD=0.19).

Finally, the best combined average score for precision (0.87, SD=0.09) and recall (0.86, SD=0.07) occurred when all sensors were used together. Compared to the best combined scores for using a single camera (pinky) with EMG + IMU, precision was only slightly worse (-0.01), but recall was significantly better (+0.19).

Comparing other cases, using all sensors significantly boosted precision (+0.13) while only slightly reducing recall (-0.04) versus only all cameras. Compared to EMG + IMU, using all sensors significantly boosted precision (+0.16) and recall (+0.10). These results show adding EMG and IMU to a camera-only model boosted precision and most importantly, adding cameras to a model that uses only EMG + IMU significantly boosted precision and recall. Given prior works [282] use only EMG and IMU to predict grasps, this result confirms the benefit of sensor fusion and shows Contextra's novel contribution of fusing multi-view cameras provides demonstrable benefits for grasp detection.

7.5.5.2 Intersection-Over-Union Analysis

In addition to precision and recall, I also include a third metric, intersection-over-union (IoU) in Figure 7.10. This metric is typically used in object recognition but I thought it would be appropriate for examining how well my model overlaps in time predictions of grasp events with ground truth labels. Typically, a value above 0.5 is considered good, however for grasp events of multiple seconds, higher is better. In my ablation analysis I saw IoUs above 0.5 when using all cameras, and EMG + IMU with one or all cameras. In general results for IoU followed the same pattern as the precision and recall ablation results with the best value (0.57, SD=0.10) achieved by fusing all sensors.

7.5.5.3 Cross-User Analysis

It is well known that EMG sensors have a high variance across users due to differences in physiology. For this reason, I also thought it would be insightful to conduct a subset of my ablation analysis on cross-user data to see the advantage of adding cameras to EMG. The results for this analysis are shown in Figure 7.11. I discussed earlier how recall was the most important metric for grasp detection to not miss true positives. In this analysis I can see clear patterns, mirroring the cross-session analysis, that show using cameras, particularly adding them to EMG + IMU sensors, helps to significantly boost recall along with all other metrics.

In cross-user analysis, I acknowledge a limitation of my evaluation on a small set of 8 participants. I believe the large volume of data (3×-20× more than related studies [50, 282]) and positive results from these participants were strong enough to show the validity of Contextra's method. However, ultimately I was not able to capture the most diverse set of possible hand physiologies and grasp poses. As such, additional testing on more users would be necessary to fully evaluate the generalizability of this approach.

Comparing Contextra to prior work, I reference the performance columns from the table in Figure 7.1. Direct comparisons between systems is difficult due to the differing performance metrics, experimental conditions, and tested objects. Nevertheless, focusing on the cross-session and cross-user cases, Contextra exceeds or matches performance of prior systems and uses what I believe to be a more challenging object set (smaller and lighter-weight).

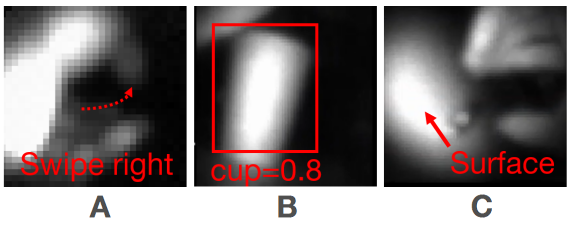


Figure 7.12: Examples of future applications for data collected from Contextra along with demonstrative example camera frames. Contextra could be used to detect micro gestures on the hand or objects (A), the identity of a particular object (B), or when a user's hand is near a surface for surface-based interactions (C).

7.5.5.4 Performance Trade-offs

For full clarity, I highlight some of the trade-offs made for this performance improvement. To recap, for power, the EMG and IMU sensors from my band collectively consume 150 mW and each camera sensor consumes 11 mW. For bandwidth, the EMG and IMU sensors collectively produce 126 kB/s of data and each camera sensor produces 66 kB/s. This means by adding cameras precision is boosted by 23% and recall by 13%, but power is increased by 22% and bandwidth by 157%. Different embedded systems may have different budgets, but I believe for the target application of using grasps as a wake gesture for contextual AI, a significant increase in recall is well worth these trade-offs.

7.6 Discussion

7.6.1 Future Work for Grasp Applications

In this subsection, I provide discussion of extended applications where Contextra could be used in the future. To recap my rationale behind focusing on grasp, I see detecting grasps as a key building block to enable deeper and more powerful contextual experiences. The ability to continuously and reliably detect when objects are picked up is what makes the following experiences possible.

7.6.1.1 All-Day Context Sensing for AI

The vision behind AI hardware devices, like the Humane AI Pin [117] and features of the Apple Vision Pro [12], is to eventually have a device that can learn about a user's behavior all day and use that knowledge to provide just-in-time contextual assistance. However, present limitations in the power consumption of such devices prohibit their runtime and privacy concerns may restrict constant use of prominent wearable cameras. Contextra on the other hand, can continuously run because of its privacy-aware and low-power sensors. By using the two devices in conjunction, Contextra can solve problems of current devices by continuously sensing at the wrist to keep up with context info, then waking up the headset's sensors for AI assistance when ready.

7.6.1.2 Using Objects to Infer Context

Using external devices in a constellation with Contextra has additional benefits for context recognition. My wristband can know how you are grasping and interacting with an object, while the headset can know what object you are interacting with. More concretely through an example: (1) Contextra detects the user is holding something, (2) a headset is woken up and its cameras are used to determine the user is holding a water bottle, (3) EMG and optical sensors from the wristband are used to determine how the user is holding the bottle: they are opening it then drinking from it. The context is the user is taking a drink of water. Once context is determined, relevant interactions can be enabled, such as having a habit app update a daily hydration tracker.

7.6.1.3 Additional Uses of Camera Data

When experimenting with the output from my camera setup, I hypothesized that it may be possible to track other interactions that build upon grasp. Examples, along with relevant camera data, are shown in Figure 7.12.

One of these is micro gestures on the fingers or held objects. Thumb gestures such as pinches, taps, and swipes are a potential source of input for a wearable that does not have traditional buttons or touch screens. Once an object is detected as grasped, Contextra may also be able to track thumb gestures on said object to make the object interactable. Returning to the banana example, if a user picks it up and performs swipes on it, they could cycle through information on a heads-up display like nutrition facts or recipes.

Other potential interactions include performing object recognition directly from the Contextra camera data or identifying micro-contexts of the hand. Such micro-contexts include if it is on a surface, in a pocket, or near the head. All together, these future directions hint at potential for Contextra's sensing stack to be taken beyond grasp and enable new context-centered experiences.

7.6.2 Limitations

I believe my proof-of-concept prototype demonstrates the feasibility of my multi-view optical fusion method for detecting grasps. Nevertheless, there are limitations to consider. First among these are characteristic limitations of wrist-based optical sensing. When placed on the wrist, cameras can occasionally suffer from occlusion. Two common [50, 58, 111, 282] techniques for offsetting this issue, namely using multiple camera views and fusing data from sensors not affected by occlusion like (EMG and acceleration), were both used in this work. Contextra also has the advantage that even if the fingers are not in view, the object may be able to be imaged. Still, in some rare cases, such as a tiny, light, object that is gently picked up with the fingers straight ahead, detection accuracy may drop.

Unique to my optical sensing technique is also sensitivity to IR backgrounds with a high dynamic range. A reminder: my camera processing uses a subtraction step between the scene illuminated by the IR LED and background IR illumination. Although I only conducted my study indoors, my sensor has an auto-gain feature, and I tested that clear images can still be obtained in sunlight. Challenging cases occur when the user is in a dark environment, which the auto-gain tunes to, but

sees a light source or a bright window in the camera view. This can produce small artifacts in the subtracted image which might be mistaken for objects. However, if a future software model was developed to perform background subtraction on raw 60 x 60 camera frames, this algorithm could also take care of these IR artifacts.

This point connects to the final issue on limitations of my camera sensors. First, my cameras are susceptible to occlusion such as long or baggy shirt sleeves. In such cases detection may still work, albeit with reduced performance, by using only EMG and IMU. Second, some objects are too small or have a surface so dark and matte that they are not well captured by the IR cameras. At their core, the camera sensors work by looking for differences in reflectivity between the objects and background that correspond to object faces and contours. As such, cases such as a dark, matte object on a highly reflective background could be recognized, but a matte object on a matte background would be more challenging. By looking at the raw 60 x 60 outputs from my image sensors, I know the features of such objects can be made out better when more pixels are available. As such, I expect that if the higher-resolution data were made available, grasp detection of these types of objects would improve, however there would be trade-offs in privacy versus my current low-resolution approach.

7.6.3 Implications for Practitioners

Many of the insights from this research may have practical implications for practitioners designing context-dependent applications. For one, I have demonstrated the ability to provide always-on context detection at a low power (255 mW) and that conserves privacy for users versus constantly running headset cameras. For practitioners this means they can develop apps that treat event detection as a black box and can focus instead on recognizing events that have been detected by my system.

My quantitative results showed that grasp detection performed best on larger objects with a larger profile to image while held in the hand. Exploration and testing of my camera sensors also showed me better optical performance on glossy objects and those with high contrast from the background. As such, designers should structure the entry points to their contextual apps around objects with these properties. For example, if tracking calories consumed from a bag of popcorn, reliability would be better for detecting grasp of the bag rather than individual kernels.

Finally, my ablation results showed that while performance was best when fusing all sensor streams, acceptable performance could still be achieved when using a subset of sensors. This is valuable information for practitioners who may want to design apps that could change their wake up expectations dynamically based on sensors available. For instance if a user puts on a sweatshirt with long sleeves that cover their wrist, practitioners could detect the cameras are blocked, switch to EMG+IMU-only mode and lower the grasp confidence threshold outputted from Contextra to wake up their app. This design would ultimately lead to more wake ups than may be needed, but would help to maintain a consistent end-user experience even when some sensors are obstructed.

7.7 Summary

In this chapter I presented Contextra, a wrist-worn system that uses sensor fusion of EMG, accelerometer, and multi-view IR camera data to detect when users grasp objects. My approach builds upon prior systems for grasp detection by incorporating a novel late-fusion model that takes event-based processing of EMG and IMU signal spikes and adds frame-based processing of low-resolution IR camera data from three different sensors positioned around the wrist. For the first time, I can detect grasp in a wrist form factor while still being privacy-aware, low-power, and enabling reliable detection across objects. Through a user study involving picking up a diverse set of objects and incorporating natural grasp poses and motions, my system yielded a grasp detection recall of 0.86. Interactions with objects form the foundation of many of the contexts I am in and activities I perform. By creating a device that is able to reliably detect object grasps, my hope is to provide a key component to enable an exciting variety of context-aware interactions.

The results I found from Contextra were strong compared to prior wrist-based grasp detection systems; however, I believed there was still room for improvement. To work reliably in the real world, a system needed not only better performance, but also to be more robust to a wider variety of grasp poses and object types than Contextra tested for. To address this, in the next chapter I will detail my work on building on a new smart ring platform that built upon Contextra's sensor fusion framework to detect hand-object interactions. But first, the next chapter will begin with a discussion about advances in AI models that can detect general-purpose activities and how a key limitation is holding back their utility for wearable devices; a limitation that Contextra and TrigRing can help solve.

Chapter 8

Context Recognition Using Low-Power Sensor Triggers

8.1 Introduction

Designers of mobile and wearable systems have long sought to infer user context and activity from continuously sensed data. Accurate context recognition enables a wide range of applications, including just-in-time assistance, memory augmentation, accessibility support, and automated logging of daily activities. Because wearable devices are worn continuously and move with the user, they are naturally positioned to support always-on context sensing. However, real-world environments are complex and dynamic, and many sensing approaches struggle to generalize reliably beyond constrained settings.

Recent advances in intelligent wearable systems have begun to shift this landscape. Egocentric cameras embedded in devices such as smart glasses and wearable "AI assistants" can now be paired with vision-language models (VLMs) to interpret user activity directly from visual input. Unlike earlier approaches that required task-specific training data and user-specific calibration, VLM-based systems benefit from large-scale pretraining on diverse internet data, enabling more generalizable, open-vocabulary understanding of user behavior. Despite these advances, continuously running camera-based VLM systems remains impractical for all-day use in real-world wearable settings. First, privacy concerns remain a major barrier: both users and bystanders often perceive always-on wearable cameras as intrusive, particularly when continuous image capture is required for inference. Second, large-scale API usage introduces significant monetary cost; even modest per-query pricing becomes prohibitive when scaled to hundreds or thousands of inferences per day. Finally, and most critically, continuous camera sensing imposes substantial power and thermal constraints on wearable devices. Existing smart glasses and similar form factors already operate under tight energy budgets, and continuous image capture and processing can quickly lead to rapid battery depletion [194] and uncomfortable device heating [117].

To address these limitations, I propose a tiered sensing framework for wearable context recognition in which high-power inference (e.g., VLM-based camera analysis) is activated only when necessary. Rather than operating the camera continuously, the system keeps it in a low-duty or inactive state and relies on a

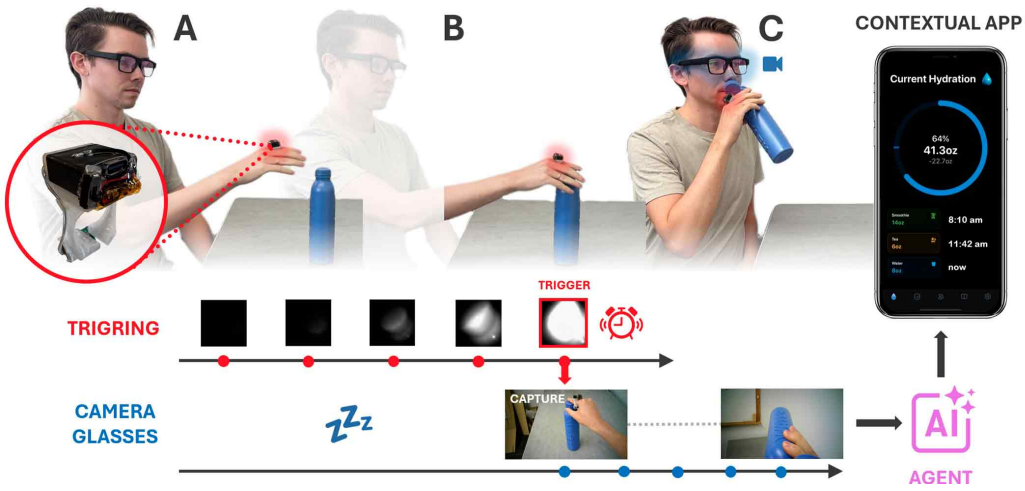


Figure 8.1: TrigRing is a smart ring system that fuses information from a suite of onboard sensors to detect when a user interacts with an object. Designed to be paired with an egocentric camera device such as smart glasses, TrigRing (call-out) keeps the glasses asleep as the user reaches for an object (A), then uses its low-power, privacy-preserving sensors to detect the grasp and trigger the glasses (B), waking them only at the moments needed to provide contextual AI assistance (C).

lightweight always-on sensing layer to detect moments of interest that warrant visual analysis. This approach reduces continuous computation, improves privacy by minimizing image capture, and lowers both energy consumption and inference cost while preserving the benefits of high-capability vision-language reasoning when it is needed.

I instantiate this framework in TrigRing, a finger-worn sensing system designed to pair with camera-enabled smart glasses. Just like Contextra, TrigRing focuses on detecting a specific and fundamental interaction signal: object grasp events. Rather than attempting to recognize full activities or general context directly, I use grasp detection as a proxy indicator that a meaningful interaction with the environment is occurring. This choice is motivated by the observation that many everyday activities involve picking up or manipulating objects, making grasps a reliable cue for when higher-level context inference should be triggered.

By detecting the onset of object grasps, TrigRing provides a low-power signal for selectively activating the glasses-based camera system, which can then perform detailed scene understanding using a VLM. This division of labor enables continuous awareness at low cost while reserving expensive visual inference for moments likely to contain semantically meaningful activity changes. I evaluate TrigRing across a diverse set of objects and interaction scenarios, demonstrating high recall of over 0.95 in grasp detection. I further show that, when integrated with a smart glasses system performing periodic VLM inference, TrigRing significantly reduces overall power consumption compared to a camera-only baseline, while maintaining timely detection of interaction-relevant moments.

In summary, this paper makes the following contributions:

- A grasp-triggered inference paradigm for egocentric context recognition, which reframes wearable sensing not as a competitor to vision-language models, but as a low-power control signal that determines when they should be invoked.

System Name	Sensor Location	Sensing Modality	Power Consumption (mW)	Weight (g)	Objects Tested	Performance (Acc. / F1)
Pistohl et al. [224]	Brain implant	Electrocorticography (ECoG)	n.r.	n.r.	1	0.920*
Xiao et al. [321]	Wrist	Force myography (FMG)	20*	40*	1	95.0%
Maufroy et al. [187]	Forearm	EMG	34	96	5	82.8%
Berlin et al. [29]	Wrist	RFID	210	35*	8	72.8%
Theiss et al. [282]	Wrist	RFID, EMG, IMU	238	128*	8	0.950 (recall)
Cofer et al. [50]	Wrist	Near-IR pair detectors	n.r.	60*	1	98.0%
Contextra [57]	Wrist	IR cameras (3), EMG, IMU	255	148	8	0.865
TrigRing (mine)	Finger	IR camera, active acoustic, IMU, lux	102	16	30	0.963

Table 8.1: Overview of key related work in wearable grasp detection. Performance metrics are derived from published materials where available. Values marked with an asterisk (*) represent my approximations based on reporting in the original text; please consult individual references for specifics.

- TrigRing, a finger-worn sensing system that detects grasp onset as a reliable interaction primitive, leveraging proximity to the point of contact to improve temporal precision of interaction detection compared to wrist- and forearm-based approaches.
- An empirical evaluation demonstrating that grasp-based triggering is accurate and can substantially reduce continuous VLM power consumption, while maintaining timely activation of vision-based context understanding.

8.2 Related Work

TrigRing is motivated by the growing use of wearable sensors for context and activity recognition. Prior work has shown that hand-object interactions are among the strongest cues for understanding user activities, leading researchers to explore sensing approaches that detect, classify, or infer interactions with everyday objects. I review three relevant areas of research: (1) egocentric activity recognition, which motivates my trigger-based architecture; (2) wearable sensing techniques for recognizing held objects and hand-object interactions; and (3) object grasp detection, the specific problem addressed by TrigRing.

8.2.1 Egocentric Activity Recognition

Egocentric activity recognition seeks to infer user activities from first-person sensor streams, most commonly video captured from head-mounted or chest-mounted cameras. A consistent finding across this literature is that interactions between the hand and surrounding objects provide some of the most informative signals for recognizing activities. Consequently, many systems explicitly model hand appearance, manipulated objects, gaze behavior, or combinations thereof.

Early work demonstrated that information about manipulated objects substantially improves activity recognition performance [126, 261]. Other approaches leveraged eye tracking to estimate user attention, using gaze as a cue to identify task-relevant objects within complex scenes [199, 219]. Similarly, hand-centric

methods have used hand segmentation, pose estimation, and hand appearance features to focus recognition on regions of the scene most relevant to ongoing activities [22, 40, 175].

Recent advances have increasingly relied on transformer architectures and vision-language models (VLMs). Compared to earlier activity-specific classifiers trained on datasets such as Ego4D and EPIC-KITCHENS [53, 87], VLM-based systems offer stronger semantic understanding and support open-vocabulary recognition of activities not explicitly observed during training [44, 52, 136, 294]. Researchers have also begun exploring multimodal approaches that combine egocentric video with inertial or physiological sensor streams to improve robustness and contextual understanding [164, 178, 179, 214].

Despite these advances, continuous camera-based activity recognition remains computationally expensive and energy intensive [34, 45, 123, 182, 200], especially compared to other sensing modalities. TrigRing is motivated by this limitation. Rather than performing activity recognition directly, my system detects moments of object interaction that can be used to selectively activate higher-power recognition systems only when meaningful context changes are likely to occur.

8.2.2 Held-Object Recognition and Hand-Object Interaction Sensing

A closely related area of research focuses on identifying the object a user is holding. Although held-object recognition differs from the grasp detection problem addressed by TrigRing, both tasks rely on sensing hand-object interactions and often employ similar sensing modalities. Consequently, prior work on held-object recognition provides valuable insight into how wearable devices can capture interaction-related signals.

8.2.2.1 Bio Signals

Researchers have explored a wide range of sensing approaches for recognizing held objects. One class of techniques measures physiological signals generated during grasping. For example, wrist-worn systems have used capacitive sensing to measure changes in wrist morphology during object manipulation [243], while others have employed electromyography (EMG) [70], mechanomyography (MMG) [242], or bioimpedance sensing to capture muscle activation and tissue dynamics associated with different grasped objects [292]. These approaches benefit from sensing directly through the body, but often exhibit substantial variation across users and sensor placement conditions.

8.2.2.2 Cameras

Vision-based approaches represent another common strategy. Cameras mounted on the wrist [58, 140, 179, 214], finger [43, 143], or head [22, 40, 65] can directly observe manipulated objects and often treat held-object recognition as a special case of object recognition [39, 113]. Because objects held in the hand typically occupy a large portion of the camera's field of view, hand-mounted cameras can simplify the perception problem relative to traditional egocentric cameras [24].

However, camera-based approaches remain computationally demanding and raise practical concerns regarding energy consumption and privacy during continuous operation.

8.2.2.3 Electromagnetic

Researchers have also explored electromagnetic (EM) sensing techniques. Passive EM systems [154] measure electromagnetic emissions naturally produced by electronic devices and motorized tools, enabling highly accurate recognition when such emissions are present [154]. Active EM approaches [292] instead inject a known signal through the body and measure how object properties affect signal propagation. These methods can capture information about an object's material composition and geometry without requiring visual observation, but often require specialized hardware and careful calibration [292].

8.2.2.4 Motion & Vibration

Mechanical sensing approaches leverage vibrations generated during hand-object interactions. Systems such as ViBand [153] use high-frequency accelerometer measurements to capture both hand motion and object-generated vibrations transmitted through the body. Other work has employed active vibration sensing [212], transmitting vibrations through an object and measuring their propagation characteristics to infer object properties. Because vibration signatures depend on material and structural characteristics, they can provide highly distinctive information about manipulated objects [151, 153, 212].

8.2.2.5 Hand Pose

Finally, a growing body of work has explored hand pose as an indirect cue for object recognition. The underlying observation is that grasp configurations are strongly correlated with object affordances and geometry. Researchers have estimated hand pose using egocentric cameras [37, 40, 70, 165], wrist-mounted cameras [58, 165], acoustic sensing systems [156, 189], and other wearable sensing modalities, then used the resulting pose information to infer object categories or interaction types. These findings suggest that hand configuration itself contains substantial information about hand-object interactions, even when the object is only partially observed.

8.2.3 Object Grasp Detection

Rather than identifying which object a user is manipulating, TrigRing focuses on detecting when an object is being grasped. This distinction is important because grasp detection generally requires less sensing and computation than full object recognition, making it more practical for always-on wearable devices. However, accurately detecting grasp events across different users, objects, and interaction styles remains challenging. Since systems for object grasp detection are most closely related to my TrigRing accessory prototype, I summarize the key techniques used in prior work in Table 8.1.

One of the earliest approaches to grasp detection relied on RFID instrumentation. These systems attached passive RFID tags to objects and used wearable RFID readers to determine when tagged objects entered close proximity to the user's hand. Early prototypes integrated readers into clothing [247], while later systems miniaturized the sensing hardware into gloves [75, 280] and wrist-worn devices [29, 74, 282]. Although effective, RFID-based approaches require prior instrumentation of objects, limiting scalability in unconstrained environments.

To avoid instrumenting objects, subsequent work explored wearable sensing techniques that directly measure grasp-related signals from the body. EMG [187, 282], force myography [321], and electrocorticography [224] have been used to detect muscle activations associated with grasping. These approaches eliminate the need for tagged objects and can provide highly informative signals, but often suffer from user-specific variability and sensitivity to sensor placement.

To address the robustness problem, several systems have combined multiple sensing modalities [50, 282]. For example, Theiss et al.'s tiered architecture paired low-power IMU motion sensing with higher-cost EMG sensing and finally RFID, activating additional sensors only when candidate interactions are detected [282]. Cofer et al.'s wristband system similarly used a near-infrared optical sensor to measure tissue displacement indicative of a grasp, then switched to an IMU to classify the grasp type [50]. The benefit of these multimodal approaches is that they can generally achieve higher robustness than single-sensor systems by exploiting complementary sources of information.

More recently, researchers have explored end-to-end sensor fusion models that jointly integrate multiple sensing streams. Contextra [57], for example, combined camera, EMG, and inertial data obtained from a wristband within a deep learning framework to detect object interactions. Compared to earlier pipeline-based approaches, this type of system can learn richer relationships between sensing modalities and often achieve improved generalization across interaction contexts.

TrigRing builds upon this progression toward multimodal wearable sensing, but differs in sensor placement and sensing objective. Whereas most prior grasp detection systems place sensors on the forearm or wrist, TrigRing senses interactions directly from the finger, placing sensors closer to the point of contact between the hand and manipulated objects. This design allows the system to capture interaction signals at their source, resulting in lower noise and fewer chances for occlusion by the hand. I believe that these improvements in grasp detection versatility move wearable systems closer to robust interaction recognition across a wide range of everyday objects. Because reliable grasp detection serves as the foundation of my continuous personal context recognition pipeline, TrigRing can function as an effective triggering accessory, enabling context-aware capabilities that were not feasible with previous approaches.

8.3 Implementation

To demonstrate an implementation of my triggering accessory framework, I created a novel smart ring prototype, TrigRing, that is paired with a set of smart

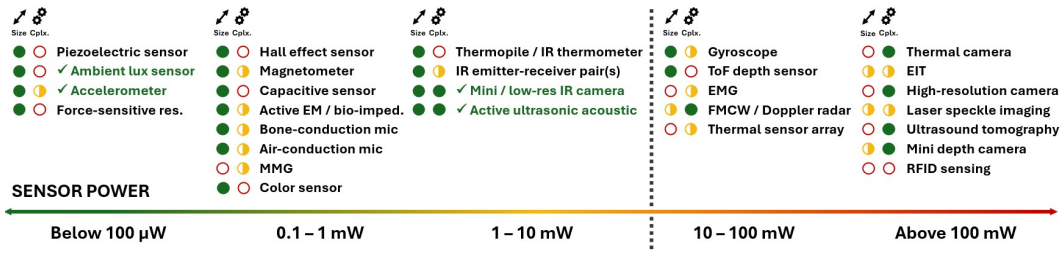


Figure 8.2: Comparison of sensing modalities for ring-based grasp detection. Sensors are grouped into order-of-magnitude power bands. Duty cycling can move some sensors down by one to two bands. **Size** rates ring-scale feasibility: ● favorable, ◐ workable with engineering effort, ○ a first-order obstacle. **Complexity** rates the richness of the raw feature the sensor delivers: ● spatially or spectrally rich (an image, spectrogram, or multi-channel pattern), ◐ a short vector or waveform, ○ a single scalar or near-scalar reading. Sensors with greater than 10 mW of power were not considered for my TrigRing prototype (shown via vertical line). A **green check ✓** indicates sensors I ultimately selected for testing.

glasses. In this section, I detail the technical background of the various hardware and software components that make up my prototype.

8.3.1 Sensor Selection

The most critical question in the design of TrigRing was determining which sensors to use. A ring is far more battery-constrained than the glasses it is meant to trigger: its form factor leaves room for only a fraction of the energy storage, yet it must sense continuously while the glasses sleep. Any sensor I selected therefore had to detect object grasps reliably while drawing orders of magnitude less power than the camera it replaces.

Based on these criteria, I created a list of viable sensors that may be useful for the task of object grasp detection. This list, shown in Figure 8.2, was constructed by looking at viable sensing techniques from prior work in detecting hand-object interactions [50, 282], as well as my expert knowledge from having developed similar wearable sensing systems in the past.

When constructing my TrigRing prototype, I aimed to incorporate as many sensors as possible. My expectation was that not all of these sensors would be equally beneficial for detecting object grasps. However, one of my goals was to perform an analysis (described in Results) examining the performance versus power tradeoff for different sensors. For example, a light sensor may perform less well than a mini camera, but if it consumes an order of magnitude less power the performance tradeoff may be worthwhile. In this way, by having a large number of sensors on my ring, I could collect all of my data at once during my user evaluation and analyze post-hoc what sensors provided the best performance. Nevertheless, some sensors, despite their performance, consumed too much power for a ring wearable. To reflect this, I decided to only include sensors with a power consumption below ~10 mW.

Based on these criteria, the final sensors I selected for my ring prototype included an APDS9500 mini IR camera, a BMI270 6-axis IMU with an accelerometer and gyroscope, a VEML7700 lux sensor, and a custom ultrasonic active acoustic

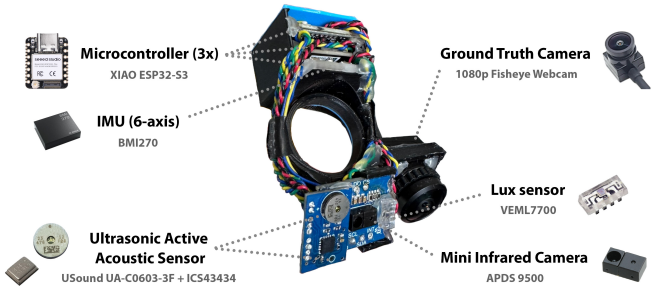


Figure 8.3: Diagram of my prototype TrigRing smart ring system (see Figure 8.11 for my final version). I take image data from a mini active IR camera and fuse it with echo signatures from an ultrasonic acoustic sensor, motion signals from a 6-axis IMU, and light readings from a lux sensor. All of this data is processed and sent over serial via three parallel microcontrollers. Finally, a fisheye webcam was mounted to the side of the ring to capture reference images to help with debugging and sensemaking of my results.

sensor created from a USound UA-C0603-3F transducer and an ICS43434 PDM microphone. Of note, the mini IR camera has only a 60 × 60 px resolution and uses an active infrared illumination technique where the sensor captures successive images with an IR LED turned on then off and subtracts them [36]. The active illumination technique acts as a form of depth filter where only parts of the scene close to the camera (and thus illuminated by the LED) are visible and parts in the distant background are removed. The mini IR camera and IMU were directly inspired by prior grasp detection systems that achieved success with similar sensors [282]. The active acoustic sensor on the other hand was inspired by prior wearable systems, unrelated to grasp detection, that have demonstrated its potential for recognizing dynamic body interactions with high-dimensional features at a low-power cost [182, 220, 338]. My custom sensor uses the same frequency-modulated continuous wave technique described in [338]. By repeatedly sending out short ultrasonic frequency sweeps and capturing their reflections, I compose echo windows and concatenate them over time into acoustic ”images” for my model (see Figure 8.4). Finally, I chose to use a lux sensor as a pseudo-baseline of a sensor that produces a simple single-dimension, stateful signal at a low power cost. My plan was to compare the performance and power cost of all other sensor combinations to the lux sensor alone.

Additional sensors that I also tested and considered were a time-of-flight IR depth sensor, an ultrasonic acoustic depth sensor, and a multi-spectral color sensor. However, I ended up not including these sensors due to high power consumption (IR depth), or high overlap with existing chosen sensors (acoustic depth and color sensor).

Many of the sensors I ended up using were the same as Contextra, however now they were positioned closer to the grasp location. My belief, grounded in many prior works, was that by placing sensors closer to the point of interaction, the information they capture will be richer, leading to higher performance. One of my goals for the project was to see how much this performance changed compared to Contextra.

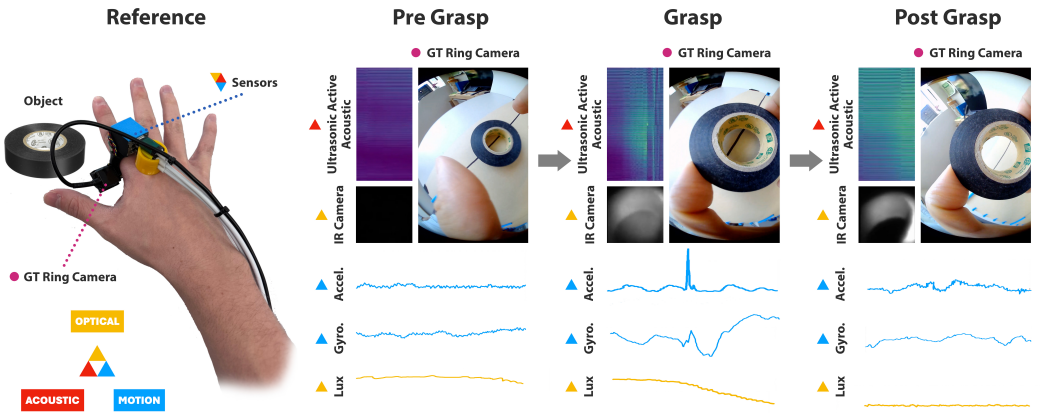


Figure 8.4: Timeline of a user grasping an object, along with demonstrative traces of sensor data from my different sources. In each frame, I highlight how different sensing channels (mini IR camera, acoustic, IMU, and lux) can contribute by sensing different features of the grasp. For example, motion of reaching for an object can be sensed by the accelerometer, then the object can be imaged by the mini IR camera when opening the grip, and finally the acoustic echo pattern changes once the object is firmly held in the hand.

8.3.2 Ring Hardware

To house my selected sensors in a smart ring form factor, I designed and created an initial prototype, shown in Figure 8.3. As best as possible, I attempted to keep a low profile, so as to not impede the movement of the fingers during grasps. However, my primary objective in designing this prototype was to incorporate all of the sensors and position them to have the best view of the grasp interaction. For this reason, I sacrificed some extra bulk to create a larger base for sensors.

The second bulge at the top of the ring contains the microcontrollers (MCUs) I used to read the sensors via I2C and send the data back to a laptop over USB serial. I chose to use the XIAO ESP32-S3 for its dual-core architecture, small size, and ease of use.

All sensors were sampled at 40 Hz or higher, which was set by the active acoustic sensor, which had the lowest maximum sampling rate (40 Hz). In testing, I discovered that even with two threads of execution and DMA-buffered I2C reads, the bandwidth of my data reads was limited by having to wait on sending or reading data while the other four sensors were being handled. As a straightforward fix, I stacked three ESP32s on top of each other and assigned one to handle the mini IR camera, one to handle the active acoustic sensor, and the last to handle the IMU and lux sensor. My hypothesis was that after evaluating performance of all of the sensors and performing ablations for sensor type and data rate, it would be possible to recommend a final design that only required one MCU.

In addition to all of the aforementioned ring hardware, I also mounted a wide-angle (160-degree), high-resolution (1080p) web camera onto the side of my ring (seen in Figure 8.3). Unlike the other sensors, this camera does not aid in grasp detection, and was never expected to be included in a final prototype. Instead, its purpose was to (1) capture ground truth image data to help aid sensemaking of collected sensor data, and (2) form the basis for a future dataset of egocentric grasp interactions. With the exception of CyclopsRing [43], I am not aware of another

Component	Power consumption (mW) @ 3.3 V
IR camera (APDS9500)	9.90 mW
Active acoustic sensor	10.21 mW
Accelerometer (BMI270)	2.26 mW
Lux sensor (VEML7700)	0.61 mW
Microcontroller (3 x ESP32-S3)	26.40 mW x 3
	Total = 102 mW

Table 8.2: Power consumption of each TrigRing component used in my first prototype. Power is dominated by the BLE stack on the MCU. With a lower data rate and sleep modes, MCU power can be as little as 1 mW. My final prototype (shown in Figure 8.11 and the Video Figure) consumes less than 25 mW.

prior work that has examined hand interactions from the perspective of a wide-angle, high-res camera on the finger. Mine would be the first to focus specifically on object grasps. I see this dataset (described again in the Open Source section) as a supplementary contribution that nevertheless may scaffold future research on grasp detection with hand-based sensing.

8.3.3 Smart Glasses Emulation Hardware

To emulate the egocentric camera on a pair of smart glasses, I also created a piece of head-worn hardware that mounted the same wide-angle, high-res web camera used on the ring to an adjustable head band. This headband was worn by participants during my User Evaluation to collect ground truth data to compare against a glasses-only system with no triggering accessory. Outside of my Evaluation, I also used the headband as a starting point to create a complete demo (described in Applications) using a pair of glasses with a USB webcam. This demo closed the loop on a complete system for detecting a grasp, triggering glasses, having the glasses take a picture, and analyzing the picture with a VLM.

8.3.4 Power Consumption Analysis

By utilizing sensors that run at a low power, one of the goals of TrigRing is to run continuously to recognize grasps where other more power-consumptive systems cannot. In this way, even when TrigRing cannot sense all parts of a user's context, it can provide a trigger to wake up other higher-power devices in a low-power sleep, such as the smart glasses, to perform deeper recognition.

Many current smart glasses have built-in cameras used for recording videos or light AI tasks. However in practice, these cameras cannot be used with any significant frequency without a substantial impact on battery life. At present, Ray-Ban Meta smart glasses are advertised to have a battery life of 8.0 hours during typical use, but in practice, when using the camera, battery life drops to around 30 minutes [76, 194]. Running and processing these cameras is a significant reason why current smart glasses do not possess "all day" battery life for AI inference tasks.

My target for TrigRing is to augment smart glasses so that they do not have to constantly run their cameras and instead could only use them when needed: allowing for AI inference, but keeping an ”all day” runtime. This is the reason I advocated for using less powerful but lower-power sensors.

Traditional high-resolution camera sensors, such as the ones I use for my ground truth, can consume in the range of hundreds of mW [68, 270]. In contrast, the mini IR camera sensor I use on my ring consumes only a fraction of the power, resulting in savings as great as 10×. The acoustic sensor I used draws slightly more power than the camera, but on the same order of magnitude. Within the acoustic sensor, the act of driving the transducer with the ultrasonic chirp consumed the bulk of the power (85%). Due to the proximity of the object targets to the transducer, it may be possible to lower the drive power (volume) of the audio amplifier. This would lead to lower power consumption, but has the potential to also slightly decrease performance. Finally the power consumed by the IMU and lux sensor was much lower, under 2.5 mW each. In particular, the lux sensor could reliably be used continuously without worry. A breakdown of the power consumption of the entire system by component is shown in Table 8.2.

8.3.5 Latency

Total latency for my system was calculated in three main parts. First, the time it took for each one of the MCUs to poll its associated sensor and receive a packet of data. For each of the four sensing channels, this latency value was different ranging from 3.7 ms (IMU) to 23.8 ms (acoustic). In addition to the polling time, there was an additional period of latency for transmitting the data over the serial connection and receiving it on the laptop. Transmission latency also varied based on the sensing channel due to the size of the packet transmitted. The acoustic channel had the largest packets, thus it likewise had the highest latency (12.9 ms). To create a complete data frame to pass as an input into my model, my system needed to wait for a new packet from all sensing channels. For this reason, the overall latency of the entire pipeline was bottlenecked by the slowest sensor (acoustic). Thus adding in the latency for model inference (24.5 ms), the total latency of the TrigRing pipeline was 61.2 ms, which corresponds to a frame rate of 16.3 FPS. Both of these values are acceptable for real-time performance in the object grasp application.

8.3.6 Ground Truth Data

To collect ground truth data during my user study, I created a customized sensing apparatus. This device, shown in Figure 8.6, consisted of pairs of infrared LEDs and TSOP1838 IR receiver modules controlled by an Arduino Mega 2560 MCU. The MCU sent 38 kHz signals to the LEDs which generated digital signals when captured by the IR receivers. Each LED-emitter pair was checked in series to avoid interference. In this way, the apparatus acted as a digital switch to detect when there was any change in the physical blockage of an IR receiver caused by an object placed directly across from it. Since I assume only one object is picked up at a time, blockage change events could be used in a simple state machine to determine if an object was picked up or placed back down.

8.3.7 Data Processing

To receive the sensor data on the laptop from the ring MCUs, I created client scripts that each ran on their own thread. Each client was assigned to read from a single serial port (assigned to one MCU) and send that data to a central server script. The server script saved the most recent packet from each client in a data buffer. Once the buffer was completely filled with new packets, the full data vector was saved to a file and the process restarted. This full data vector, which was saved to a file at a rate of 40 Hz, contained values for each ring sensor, a value for the current ground truth state, and captured frames from the ring and glasses webcams.

For data collection, such as in my User Evaluation, this is where the pipeline ended. When preparing data for model training and inference, a few additional steps were used. When provided as input to my ML model, one IR camera frame is provided per data sample. For all other sensors, a short window of data is provided for each sample. In each window of size N, the first entry corresponds to the data buffer from the most recent sample, and the rest of the window corresponds to the previous $N-1$ samples. I chose this approach because while a single IR image can capture stateful information about a grasp (e.g., can an object be seen), the other sensors capture events that take place over time. For example, the accelerometer capturing vibrations on touch contact or Doppler shifts in the acoustic echoes when approaching an object.

Under this approach I used a window size of 21 samples for the acoustic data, and 71 samples for the IMU data and lux sensor. These sizes were chosen based off of the sampling rates of the respective sensors, and my desire to have windows that were just long enough to capture the major components of a grasp, which typically last under 400 ms [321]. When preparing data from my Evaluation for training, I used a script to format the sensor data which was previously saved as a continuous stream into discrete windows. With the exception of a filter to remove the gravity component from the accelerometer data, no additional filtering or preprocessing was applied to the windowed data before passing it into my model.

8.3.8 Machine Learning Architecture

A diagram of my complete system architecture can be seen in Figure 8.5. To reiterate, TrigRing uses sensor fusion to recognize object grasps. In my model, I use a late fusion technique where I individually process each of the four sensor streams in separate branches of the model, then combine these branches before a final layer of processing. The overall architecture was inspired by the late-fusion model used successfully in Yeon et al.'s WatchHAR [331].

Each sensor stream is first processed independently, in its own branch of the model, to encode features specific to that modality. I chose separate branches because each sensor captures grasp-related information in a very different form: some signals are complex, such as the Doppler ripples that appear in ultrasonic acoustic images as a hand approaches a surface, while others are simple, such as the drop in lux sensor readings when an object is picked up and covers the sensor. To account for this variance, each branch uses a sub-architecture suited to its input. For the IR camera images and ultrasonic acoustic "images," I used a MobileNetV3 backbone [110] pretrained on ImageNet-1k. MobileNetV3 combines in-

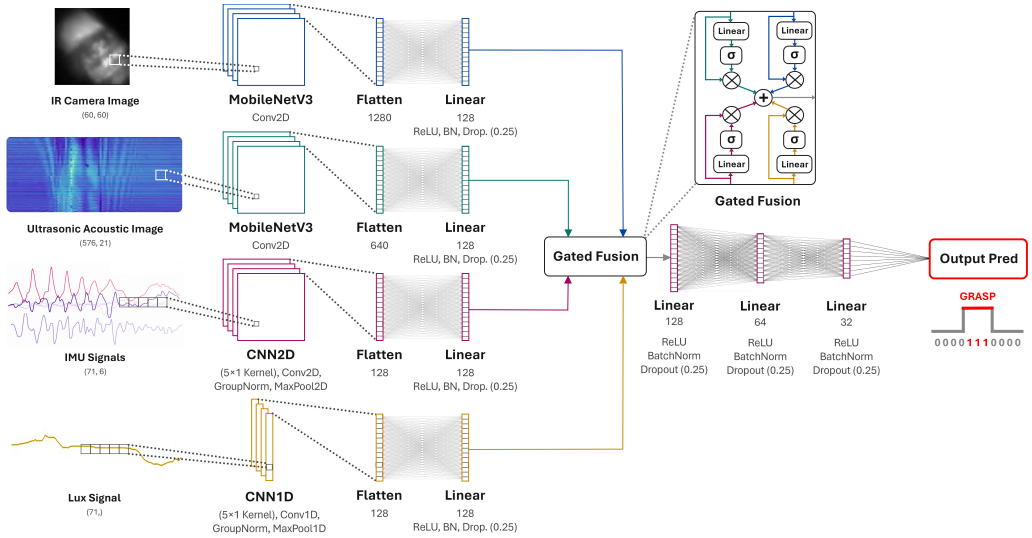


Figure 8.5: A diagram of my ML model architecture. My model was split into four parallel branches, one for each sensing channel. Each branch used a different model for featurization of raw data before passing in the extracted features into a gated fusion block which learns intelligent combinations of the sensing channels. Finally, the output of the fusion block is passed through final linear layers to make an output grasp prediction.

verted residual blocks with squeeze-and-excitation modules, platform-aware neural architecture search, and an efficient classification head, together offering a strong balance of model size, computational efficiency, and accuracy.

For the IMU and lux encoders, I adopted the ConvBoost architecture [252], a 3-layer 2D CNN designed for efficient processing of multivariate time series data such as IMU signals. Because the lux stream is single-variable, I used a 1D CNN for it instead. Both use 5×1 kernels to extract temporal features, with max pooling after the first two convolutional layers and ReLU activations throughout. The classifier head consists of two fully connected layers with dropout ($p = 0.5$) to prevent overfitting. This architecture favors simplicity and computational efficiency while still extracting effective features for grasp detection.

Rather than simply concatenating features across my four sensing modalities, I combine them using a Gated Fusion mechanism [18]. Each stream's embedding is first projected into a shared 128-dimensional space via a separate linear layer, then passed through its own gating network (a linear layer followed by a sigmoid) that outputs a weight between 0 and 1 for each feature dimension. Multiplying each projected embedding by its gate values lets the model scale features according to their learned importance before the streams are merged. The gated outputs are then summed and passed through an additional linear layer to produce the final 128-dimensional fused representation, which feeds into the classifier head. I trained the model as a regression, thresholding its floating-point output with a tunable hyperparameter to predict grasp class probabilities. Every component is fully differentiable and trained jointly, end-to-end, using binary cross-entropy loss, so the model learns to weight each modality according to its relevance to the current state (grasp or no grasp). This approach is particularly well suited

to settings where some data from the current user is available during training, allowing the gating to adapt to that user's characteristics.

8.3.9 Model Training Protocol

In total, TrigRing's model has 17 giga FLOPs and 3.1 M parameters. During testing, I found my model's performance improved if I included a larger corpus of pilot data during training. This pilot data was separate from the main training data collected on participants during the study. The size of this pilot data was roughly 1 hour of grasp events collected on data from one of the authors on a broader set of objects. My model was trained for 30 epochs on the combined pilot and user data using the ADAM optimizer and a learning rate of 1e-5. When tested on a Core i7-10750H laptop CPU, the model achieved an inference time of 24.5 ms. For reference, new frames arrive only every 36.7 ms which still gives a 12.2 ms buffer for real-time inference (which happens at 16.3 FPS). While feature extraction and model inference could eventually happen on-device, today's expectation is that data is transmitted wirelessly from the ring to a nearby phone or glasses for processing.

8.3.10 Context Recognition Demo

To demonstrate the full execution of my system and to highlight the core contribution of TrigRing, I created an end-to-end context recognition demo. This demo used my final ring prototype (shown in Figure 8.11) and smart glasses USB camera, different from my User Evaluation. This final prototype operates on a 105 mAh battery and can send data to computer wirelessly via Bluetooth. I did not run an evaluation on the context recognition accuracy of this demo, as this primarily tests the VLM which is not the contribution of this work. Nevertheless, I believed it useful to create an end-to-end demo to showcase what a complete implementation of my triggering framework would look like in action.

In the demo, the camera on the glasses was connected to the same laptop that the ring sensor data was sent to. On the laptop I ran grasp detection inference using a version of my ML model trained on all participant data from my User Evaluation. Whenever a grasp was detected, I took a picture using the glasses camera, then passed this image to a VLM (Gemini 3.5 Flash-Lite) with the following prompt: *"Analyze this image in detail and list the single object that is being held by the right hand in the image. There is a measurement sensor being worn on the index finger, ignore it. Just state the held object and nothing else. If no object is held, explicitly state 'nothing'."* The resulting context description was then presented visually for the user. A demonstration of this demo with various prompts can be seen in my Video Figure.

8.4 Evaluation

To evaluate the performance of TrigRing, I conducted a user study on grasp detection accuracy. The purpose of this evaluation was to collect user data to answer the following questions, analyzed in the subsequent Results section:

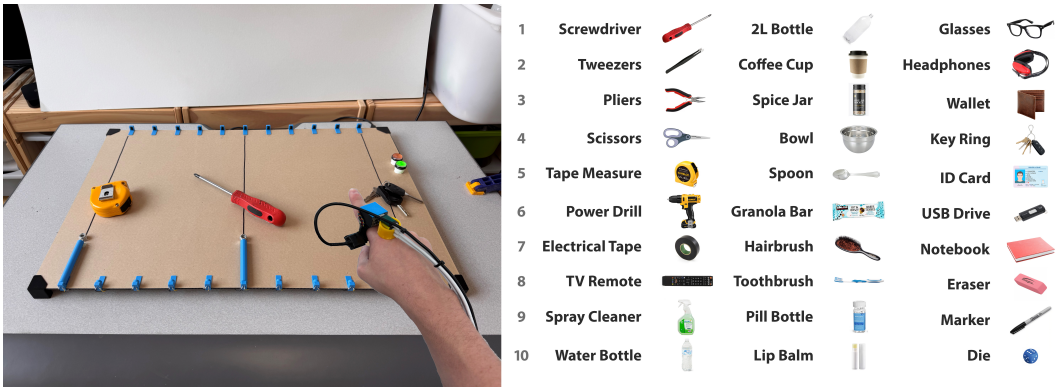


Figure 8.6: Setup for my user evaluation. During the study, a set of 30 everyday objects (shown on right) were placed in sets of three on a sensing platform (shown on left) that used infrared detector pairs to collect ground truth data on when objects were picked up and set back down. Objects were chosen in part to parallel prior work and in part to offer a diversity of sizes, weights, and reflectances.

(RQ1) How well does TrigRing recognize object grasps?

(RQ2) How does performance vary across different types of objects?

(RQ3) Which sensors aid the most in detecting grasps, and how much power do they trade-off?

I recruited a set of 18 participants (13 female, 5 male, mean age 23) for my study. The study lasted 45 minutes in length and was broken up into two identical sessions. Each session began with participants putting on the ring and glasses hardware. The same ring prototype was used for all participants. To account for differences in finger size, the base ring was sized to 20 mm, and 3D printed TPU shims were used to accommodate smaller ring sizes.

Grasp detection was evaluated on a set of 30 commonplace objects, shown in Figure 8.6. The core of this object set was taken from prior work on grasp detection [29, 57, 282] to produce an accurate comparison. The remainder of the object set was filled out by inspiration from prior work on held-object recognition [151] and other objects I believe are commonplace, yet have diverse physical properties, including size, shape, and weight.

After a brief orientation, the study began with participants standing in front of a desk on top of which was placed my ground truth (GT) apparatus. The study proctor placed three objects on top of the GT apparatus. Participants were instructed to pick up the leftmost object, hold it for one second, set it back down, then wait one second. To standardize pacing, a flashing light on the GT apparatus was used to let participants know when one second had passed after each GT state transition. This process was repeated for the middle object, then finally the right object to complete one cycle. After six cycles were completed, the proctor paused data collection and swapped in three new objects. This process was repeated (10 times) until all 30 objects were used. In total, by following this procedure, I collected data on 18 participants $\times$ 2 sessions $\times$ 6 cycles $\times$ 30 objects $=$ 6480 unique object grasps.

At the conclusion of a session, participants performed a brief sequence of five non-grasp poses, repeated for six cycles, following the proctor's verbal instructions. The poses consisted of *arms crossed*, *hand down by side*, *fist*, *fidgeting fingers*, and *hands clasped*. The purpose of this section was to collect data on the null state (no grasps) that was more diverse than the hand held in midair and more typical of other hand interactions TrigRing might encounter in real-world use.

Between each session participants took a short break and were asked to take off the hardware before re-wearing it for the second session. The purpose of this was to simulate real-world use where the hardware may shift slightly in position or orientation between wearing sessions.

As a point of clarification, TrigRing is trained to predict moments when objects are lifted off and set back down on a surface, as this is how my ground truth is configured. I note this is distinct from behavior where a user touches an object, but does not actually pick it up. This is not a case I explicitly tested for or designed my Evaluation around; however in anecdotal testing of my trained model, I observed this case overwhelmingly recognized as a grasp.

8.5 Results & Discussion

My system prototype and evaluation procedure were designed to investigate several important metrics of my system's performance. Before discussing my results, I detail my method for calculating system performance, and the process of tuning my pipeline's major hyperparameters to achieve the best and most robust performance. For my Evaluation results, I first analyze grasp detection results to compare the cross-session case, when including data from a session of the current participant, and the cross-user case, where no data from the current participant was used during training. Second and most importantly, I ran an ablation study to identify how different combinations of sensors used in sensor fusion performed and find out the relative performance contribution of each sensor channel. Finally, using my ablation results and data from my system power analysis (Table 8.2), I ran an analysis to measure and report the performance-power tradeoff for different sensor combinations.

8.5.1 Calculation of System Performance

For all analyses, results are presented in precision and recall, which I calculated from continuous model predictions using the procedure outlined in Cofer et al. [50]. I chose this approach because my model continuously outputs predictions for a grasp state while my objective is to detect finite grasp events in time, thus measuring accuracy on a frame-by-frame basis is less useful.

In my accessory triggering framework, the most important metric is recall, i.e., what percentage of the time am I correctly detecting (and not missing) true positive object grasps. The cost of missing a real grasp is high because it means the glasses will never be triggered to wake up and recognize an event that may be critical, such as taking a daily medication. On the other hand, while I still highly value precision, i.e., what percentage of the time did I not trigger on a false positive, this metric is slightly less important. This is because even if TrigRing triggers and

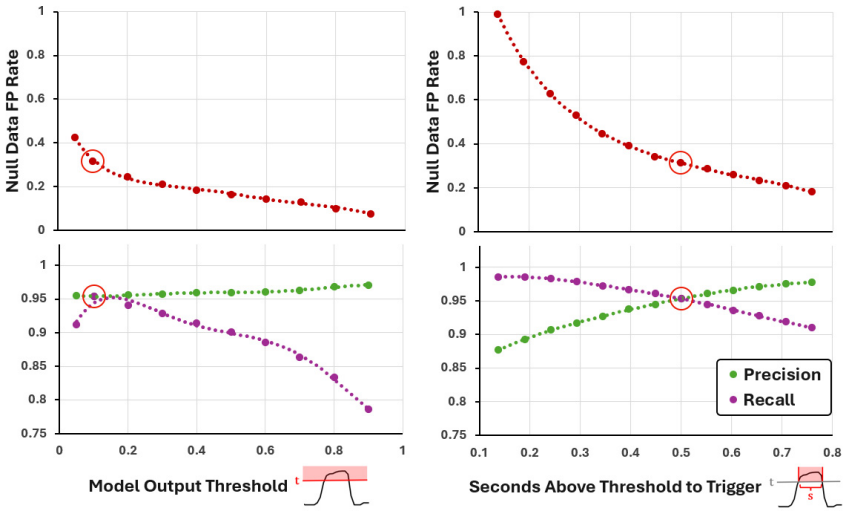


Figure 8.7: Plots detailing my pipeline hyperparameter tuning process. I tuned both the threshold for predicting a positive grasp class from the output of my model (left) and the number of seconds this positive class must be held to trigger a grasp event to the smart glasses (right). The final value for both hyperparameters (shown with the red circle) was selected as the value that maximized F1-score (i.e., maximally balancing precision and recall). Notable is that while practical performance of TrigRing is best when optimizing for recall, this significantly increases the null data false positive (FP) rate, thus I optimized F1-score instead.

wakes up the glasses on a false positive, the worst case scenario is that the glasses recognize nothing is held in the hand and go back to sleep, wasting a little power. To maintain practicality, precision needs to remain high, otherwise there will be too many false positive triggers and battery life will plummet, however the cost of missing a real grasp is much graver.

Finally, as a separate analysis I also include the false positive rate on the null evaluation data. While precision and recall illustrate TrigRing's performance in correctly recognizing intended grasp events, the null false positive rate shows how often the system incorrectly fired a trigger (and thus wasting power) when participants performed one of the instructed null gestures.

8.5.2 Model Hyperparameter Tuning

Before calculating metrics on my model's outputs, I first pass the predictions through a filter stage. The first step is to take my model's regression output and apply a threshold which determines whether the prediction is under (no grasp) or above (grasp). I then apply a step that looks for short transitions between the low (no grasp) and high (grasp) states. If any section above the threshold lasts less than 0.5 s, I determine it to be spurious and filter it out. Any grasp prediction longer than 0.5 s is left alone and considered a valid trigger. This filter stage cleans up noisy oscillations in my output predictions.

Both the model output threshold and the seconds above the threshold used to fire a trigger were considered hyperparameters. A detailed analysis of my process for tuning these hyperparameters is shown in Figure 8.7. The tuning process was

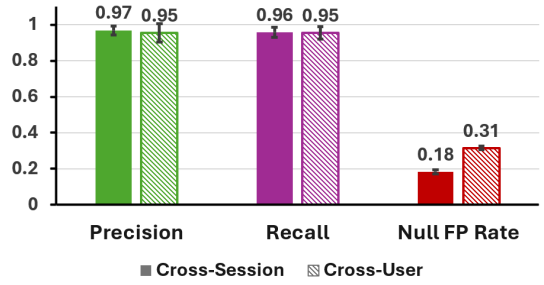


Figure 8.8: Averaged precision, recall, and the null false positive (FP) rate across participant trials for the cross-session and cross-user cases. Precision was slightly higher and null FP rate lower (both good) in the cross-session case due to the inclusion of distinctive training data from that participant in the analysis.

conducted on the cross-user case for my three primary metrics: precision, recall, and the null false positive rate.

8.5.3 Comparing Cross-Session and Cross-User Results

The results for my first analysis, which compared the performance of TrigRing for the cross-session and cross-user cases, are shown in Figure 8.8. The metrics were calculated by running the relevant session data for each participant through a model trained for that case (cross-session or cross-user). For the cross-user case I used a model trained on data from all other participants (including pilot) except the one under test. I took this trained model then tested it on the data from both sessions for that participant and calculated metrics. This process was repeated with a new trained model for each participant, then results were averaged across all 18 participants.

The process for the cross-session case was identical, except for training I took the cross-user model for that participant and fine-tuned it on the data from one of that participant's sessions. This trained model was then tested on the other session and metrics calculated. This process was repeated for the other session, then results were averaged across both sessions for all 18 participants. My expectation was that performance would be better for the cross-session case, as the model had access to some training data from the current user. This allowed the model to gain some understanding about how the user grasps objects or performs null gestures.

8.5.3.1 Grasp Detection Results

Final grasp detection results between the two cases were nearly identical. TrigRing achieved strong grasp detection performance not just in cross-session, but in both cases, with a precision and recall of over 0.95, which is significantly better than similar prior works which detected object grasps from the wrist [29, 50, 282]. In addition, these results are also a significant improvement on the most similar prior work, Contextra [57], which achieved an average precision and recall of 0.82 and 0.84 for cross-user. I believe that these improved results are a consequence of the sensing modalities I chose, my powerful fusion pipeline, and perhaps most

critically, the advantage of sensing grasps from a finger perspective. Earlier in the Implementation section I mentioned how sensing hand behaviors closer to the point of interaction can result in higher quality and more informative signals. I believe the results for TrigRing's ring-based approach and comparing to previous wrist-based work provide evidence for this advantage.

Looking in more detail, the cross-session case achieved a precision of 0.973 (SD=0.02) and a recall of 0.957 (SD=0.03). In comparison, the cross-user case achieved a precision of 0.956 (SD=0.01) and a recall of 0.951 (SD=0.02). Between the two cases, the results were within a statistical margin of error of each other, with the cross-user case only 1.7% lower than the cross-session case for precision and 0.6% for recall.

8.5.3.2 Null Data Results

Null data results between the two cases were more stratified. These results were calculated exclusively on the 30 null gestures that participants performed each session. This data was included in this analysis but left out of all other analyses. I designed the five null gestures (*arms crossed*, *hand down by side*, *fist*, *fidgeting fingers*, and *hands clasped*) to be gestures users might plausibly perform in daily activities, but also adversarial in the sense that they might produce sensor data with a high degree of similarity to an object grasp.

To quantify null data performance, I calculated the rate that TrigRing triggered a false positive on each of the null gestures for each participant and averaged the results. This produced a null false positive (FP) rate for the cross-session and cross-user cases, shown in Figure 8.8, where a lower rate is better. The cross-session case had a null FP rate of 0.18 (SD=0.02). In comparison, the cross-user case had a null FP rate of 0.31 (SD=0.01) which was 72.2% higher.

While the cross-session null FP rate is acceptable, there is some room for improvement in the cross-user case to ensure that the combined ring-glasses system consumes a practical amount of power. Ideally, the null FP rate should be under 0.2. One known solution for improving the null FP rate is to include more training data from the current user. The amount of null gesture training data that I collected for each participant was also six times less than the amount of grasp training data. This meant not only that there was an imbalance in the training data, but also that there were far fewer null gesture examples to learn from. In line with the performance improvement from cross-user to cross-session, I hypothesize that with access to more null gesture examples, null performance could improve.

8.5.4 Sensor Ablation

The goal of my second analysis was to determine which sensing channels contributed the most to recognizing grasps. I did this by conducting an ablation on my cross-user results. For each participant, I took the model trained from the cross-user analysis and disconnected some of the connections in the final addition of the gated fusion block based on which sensors I wanted to ablate. The results from testing were averaged across all participants and this process repeated for every possible combination of sensor channels. The results from this ablation are

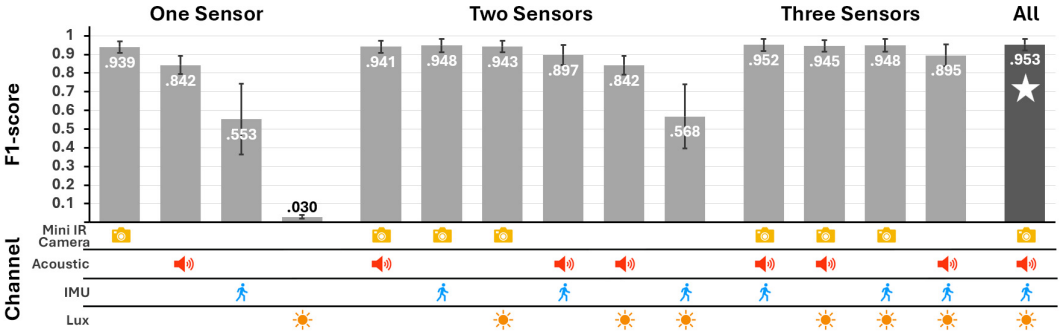


Figure 8.9: Results for my ablation analysis of every possible sensor combination. Colored icons on the bottom indicate which sensor channels were used for that column. The best combination (all sensors) is marked with a star.

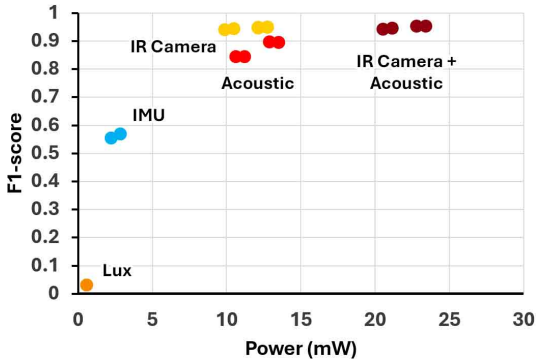


Figure 8.10: Comparison of the performance of different ablation sensor combinations and the total power consumption of the corresponding sensors for that combination. This comparison shows that the best combinations to maximize performance and minimize power use the mini IR camera, but not the acoustic sensor.

shown in Figure 8.9. To improve clarity of the figure, I chose to plot F1-score as a combined measure balancing both precision and recall.

Among all sensor configurations, the best performance, F1-score 0.953 (SD=0.013) was achieved when all sensing channels were used, which is the default configuration used in all other analyses. Performance tended to drop off lower as more sensing channels were removed with three sensors achieving the next best results, then two sensors, and finally one sensor the worst. This result matches what would typically be expected of sensor fusion where more sensing channels unlock better performance by allowing the model to see more unique featurizations of a grasp.

However, notably in TrigRing's ablation results, while performance dropped when using fewer sensors, it still remained very high when using just the mini IR camera or acoustic sensors. In fact, compared to the all sensors baseline, the just mini IR camera achieved an F1-score of 0.939 (SD=0.021) which is only 2.5% lower, and the acoustic achieved an F1-score of 0.842 (SD=0.045) which is 11.6% lower. The sensing power of the mini IR camera and acoustic sensors can all be seen in the conditions with two or three sensors where any condition that included one of these sensors performed at worst within 11.6% of the baseline.



Figure 8.11: Photos of my final proof-of-concept ring after my power analysis. Compared to my earlier prototype, this ring has nearly identical performance, but consumes less than 25% of the power, runs wirelessly on a battery, and is significantly slimmer.

8.5.5 Performance-Power Tradeoff

My prototype sensing ring was intentionally designed with a number of sensors and microcontrollers with the understanding that not all of them would help equally well with sensing grasps and while some may work well, others may not be worth including in a final prototype. This assumption turned out to be correct as the camera and acoustic sensors were significantly more helpful than the IMU and lux sensors.

Overall, the ablation results indicate that although better performance is achieved when more sensors are fused together, nearly comparable performance can be had when only one sensor is used. This has significant implications for the practical deployment of TrigRing. Each additional sensor used incurs a power consumption cost and the secondary goal of TrigRing beyond detecting grasps is to consume as little power as possible.

To quantify this performance-power tradeoff I plotted the F1-score performance of all 16 sensor combinations against the respective power consumption of the corresponding sensors. These results are shown in Figure 8.10. The main finding from this analysis was that the best sensor combinations to maximize performance while also minimizing power include the mini IR camera but leave out the acoustic sensor. This is because first, the mini IR camera achieves very strong performance on its own, much higher than the IMU, and thus is worth spending more power for. On the other hand, adding in the acoustic sensor increases power consumption by 103%, but leads to an only 0.2% improvement in performance, which is well within a margin of error.

These results led me to conclude that the optimal TrigRing prototype could include only a mini IR camera sensor, and optionally an IMU and lux sensor if 1% better performance is desired for a small increase in power consumption. In accordance with this conclusion, I created a second TrigRing prototype (shown in Figure 8.11), which achieves similarly strong performance as my original prototype while consuming significantly less power and having a notably slimmer profile. Both lower power consumption and a slimmer profile help to make TrigRing more practical for real-world use. The design and performance of the final prototype is shown in action in my Figure 8.12 Video Figure.

8.6 Applications

The purpose of the TrigRing prototype is to demonstrate a prototype implementation of the triggering accessory framework for continuous context recognition. In the Results just presented, I demonstrated the viability of using a fusion of sensors on the finger to detect object grasp triggers. The realized benefit of a robust and accurate triggering accessory is that it enables the general-purpose context recognition of the smart glasses, powered by a VLM, to run "continuously". Continuous context recognition in turn opens up the possibility of many practical end-user applications, which I show in Figure 8.12 and detail here.

8.6.1 Habit & Dietary Logging

One of the reasons habits can be notoriously difficult to keep and keep track of is that they require persistent upkeep, particularly remembering to perform or log certain behaviors. With the introduction of a device that can automatically detect the moment habits do (or should) occur, the burden of having to remember to log every instance is offloaded.

In this way, my system could help users to automatically track and log many habits, such as flossing or watering plants. In this example, TrigRing would expect to detect flossing once per day, but if this did not occur, my system would know the habit was missed. In a similar fashion, automatic logging is also useful for enabling automatic dietary monitoring, such as tracking what times of days meals are eaten, what type of food, and how many calories consumed. In prior research, specialized wearables were developed specifically for this task [26,27,170], in contrast, TrigRing can enable it with a general-purpose device.

8.6.2 Medication Adherence

Another important application, which is related to habit monitoring, is tracking adherence to taking medications. In the United States, approximately 40% of prescribed medications are taken incorrectly or not at all [206], which are linked to 125,000 avoidable deaths every year [5]. If given prior information about what medications a user should take and when, TrigRing can create a record of when medications are taken and send timely reminders when they are missed.

8.6.3 Guided Task Completion

During physical tasks that are complex and require multiple steps, such as cooking a recipe, assembling a piece of furniture, or fixing a leaky faucet, users can greatly benefit from automated assistance that recognizes what step they are on and what to do next [73, 146]. While traditional voice commands can be used to prompt smart glasses about the next step, this can be tedious and may require additional prompts to remind the system what step you are on. Instead, TrigRing could track the moments objects are picked up and interacted with and use those triggers as cues to tell the system to look for the next step. This removes the need for manual prompting and still allows the camera-VLM system to not be constantly running, which is draining for longer tasks.

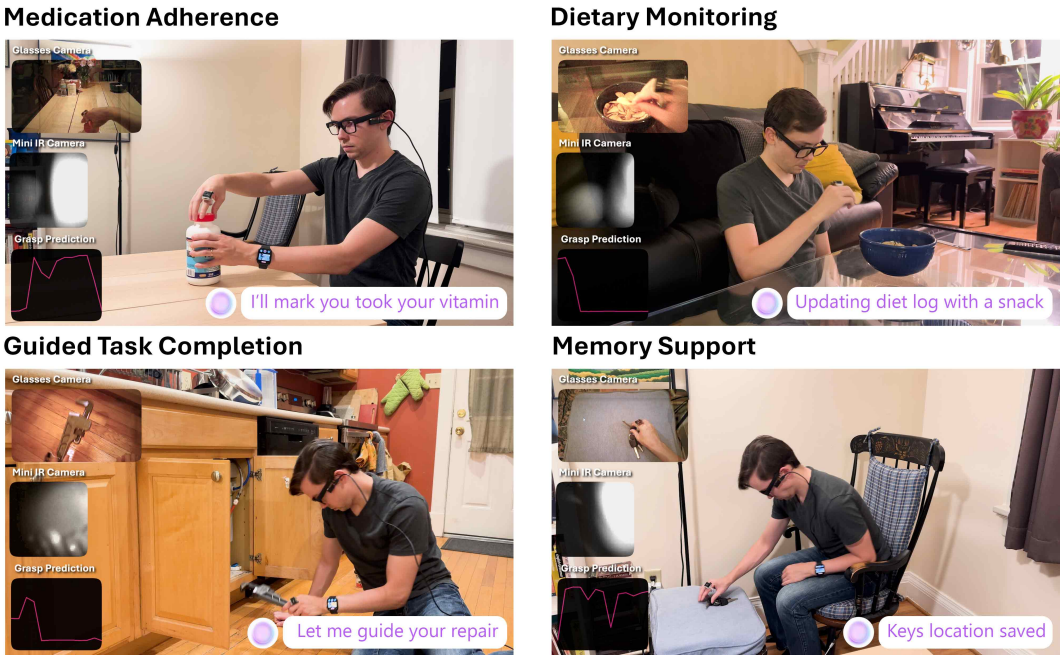


Figure 8.12: Example applications that TrigRing enables when paired with camera-equipped smart glasses. TrigRing tells the glasses when to capture an image and send it to an AI agent, which can then help with tasks such as logging that you took a medication (top left), tracking your diet (top right), guiding you through a DIY repair (bottom left), and remembering where you left your keys (bottom right).

8.6.4 Memory Support

Another application is augmenting the spatial memory of users. It is a well-recognized problem that users may misplace objects that are important to them, such as keys or a wallet, and seek assistance in locating them. Many commercial solutions already exist for this problem such as AirTags or similar devices [8, 162]. However, instrumenting every device you want to locate can be costly, and also adds bulk to the objects. In addition, in some cases a user may not realize they want to track an object until they have already misplaced it. This is especially true of users with memory impairments such as dementia.

To solve this problem, TrigRing can track instances when users leave behind objects (i.e., the end of an object grasp) that are important to them and keep a log of their last known locations. Then, if the user wants to know the location of a particular object in the future, all they have to do is ask their glasses where they last put it. With this method, the upfront cost of instrumenting objects is removed. In addition, if combined with VLM scene understanding, the user can also ask questions using natural language, such as ”where are my keys?” or ”what was I doing when I forgot my ID card?”. Without implicit input, this type of continuous context understanding is not possible with current explicit AI interactions.

8.6.5 Dictation for Low-Vision Users

With current camera-based dictation software, users must manually take a picture or continuously capture video to get audible read outs of text that is visible in front of them. This works well for short tasks, but during longer tasks, continuous video can drain battery. In addition, the user may only want text read under certain conditions, such as ”read the sender whenever I pick up an envelope” and ignore all other text. With object-based continuous context recognition, TrigRing could make this kind of application possible.

Further, in a manner more reminiscent of a personal assistant, a TrigRing-glasses system could learn over time when and where a user wants dictation assistance and know to automatically step in at only those moments. For example, learning to begin dictation whenever the system detects the user opens the refrigerator.

8.6.6 Shopping Assistant

Finally, TrigRing could help serve as a personal assistant for in-person shopping. When shopping online, many tools and plugins are available that allow customers to quickly and easily compare prices and availability of items to competitor stores. Currently, there is not a good analogue for such plugins when shopping at physical stores. Smartphone apps exist that allow customers to compare items in the manner of internet shopping, but this can be cumbersome and slows shoppers down by requiring a great deal of manual input.

In contrast, when used in a store, TrigRing can automatically bring up price comparisons for users when they pick up items, or provide other information such as availability of other colors or models at a different store. Critically, all of this assistance can happen automatically whenever an item is picked up without the user having to explicitly ask for help every time.

8.7 Limitations & Future Work

As with any method, TrigRing has strengths and weaknesses. I now highlight significant limitations and potential avenues for future work.

8.7.1 Additional Worn Device

The most obvious limitation of TrigRing and the triggering accessory framework is that they require the user to wear an additional device beyond the smart glasses (or similar camera-based wearable). However, there are several counterarguments to this.

For one, beyond its use in detecting object grasps, a smart ring could also be used to provide more traditional forms of explicit input, such as selection, directional, or pointing input. Basic explicit input is a problem that current smart glasses already struggle with due to their compact size on the face and lack of ways to sense the hands. As a result, many commercial smart glasses solutions already give users the option to wear an additional input device.

In addition, while the results I have presented and sensors explored are only applicable for a smart ring, the idea of a triggering framework need not be limited to a ring. For example, instead of using sensors in a second device, low-power sensors could be integrated into the same device (smart glasses or otherwise) that has the egocentric camera. Since egocentric cameras require a view of the entire body, this device would likely not be on the hand, which introduces challenges for sensing grasps. Nevertheless, I believe sensors such as the mini IR camera and ultrasonic acoustic sensor could be used even at a distance.

8.7.2 Grasps with Non-Worn Hand

A classic limitation of wearables worn on the hand, such as a smart ring, is that they are only designed to sense interactions by the hand they are worn on. TrigRing has this same limitation. This means if an object is grabbed with the non-worn hand, it will not be detected and will not generate a trigger.

One straightforward solution to this problem is to have the user wear a second ring on the other hand. This introduces extra cost to the user, but does have other advantages, such as creating recognition redundancy when the user picks up objects with both hands. Another future solution could also get around this problem by embedding the triggering sensors in a wearable that has a better view of both hands, such as the smart glasses themselves.

8.7.3 Non-Standard Grasps

During my user study, users were not given any specific prompts about what grip or style to use when picking up objects. As such, the results I presented represent performance when users naturalistically pick up objects.

Nevertheless, in testing I became aware that there are some non-standard grasp styles which can make object grasp detection more difficult. For example, since TrigRing is designed to be worn on the index finger, grasps that omit this finger, such as pinching the pinky finger and thumb, may lead to worse performance. This is because the sensors I used are all positioned on the bottom part of the ring, and thus come into close contact with the object during a grasp with the index finger. If the index finger is not used, the objects may not appear in the field of view of the IR camera or light sensor.

However, despite this challenge, anecdotally I did go back to my user study data and found examples where participants used some of these non-standard grasps. While the optical sensors did not image the objects, performance still remained acceptable. I believe this is because sensors such as the ultrasonic acoustic sensor and IMU still behaved as normal, highlighting a key benefit of using a sensor fusion approach as I did.

8.7.4 Complex Grasps & Environments

In designing my user evaluation, I primarily focused on including a diverse set of objects and replicating a similar protocol to prior grasp detection experiments [29, 57, 282]. I believe this presents a strong initial measure of grasp detection

performance, however it did not include testing on more complex grasps and environments that may appear during use in the real world.

For example, in a more complex scenario with non-static objects, a user might pick up an object then manipulate, morph, or deform that object while holding it. When picking up the object, there may also be more environments with dense clutter, moving objects, etc. I did not explicitly test any of these more challenging scenarios, and thus cannot precisely say how performance might vary, but I believe it does call into question what should be considered a grasp.

Two more complex scenarios I did observe in my evaluation data included picking up an object and passing it between one's hands and touching an object without picking it up. Although these were not labeled as positive examples in my ground truth, TrigRing tended to identify them as grasps. At a high-level I find this behavior acceptable as it would be more costly to miss grasp-adjacent events a user might want to detect as opposed to triggering more often and going back to sleep if not needed.

8.7.5 Form Factor

I also note that TrigRing's smart ring prototype is very much a proof-of-concept implementation, and several engineering challenges would have to be overcome in a future consumer version. Chiefly, the form factor of my ring prototype is bulkier than what users typically might expect out of current commercial smart ring offerings [103]. One way size could be reduced is to use a more powerful or specialized processor that can handle more sensor streams at once; this would reduce the need to have three MCUs like I used. Based on the results of my performance-power analysis in Section 8.5, future designers may also choose to leave some sensors out which could further shrink the form factor.

8.8 Summary

In this chapter, I presented my work on a novel system for using a triggering accessory to enable continuous personal context recognition. To practically demonstrate this concept, I created and evaluated TrigRing, a smart ring that fuses low-power sensors to detect object grasps and uses these grasps as triggers for smart glasses. By combining a mini IR camera, an ultrasonic acoustic sensor, an IMU, and a light sensor, TrigRing detects grasps with a high degree of accuracy: in a user study across a diverse set of objects, it achieved 0.95 precision and recall, and I outlined the tradeoff in performance each sensor contributes relative to its power cost. TrigRing completes my investigation into three promising avenues for sensing implicit hand input — touch, pose, and object grasps — and I believe it demonstrates a powerful technology for enabling truly continuous context recognition, along with a suite of context-centered applications that were not previously possible.

In the remainder of this thesis, I present my overall findings: first, recommendations for future wearable sensing devices (Chapter 9), then a discussion of this thesis's contributions and avenues for future work (Chapter 10).

Chapter 9

Design Recommendations for Wearable Hand Sensing

My journey of building the systems in this thesis involved repeated decisions about which sensors to use and how to combine them. To aid future designers and researchers, this section distills those decisions into recommendations: the criteria I weighed, the single sensor and sensor combination I would recommend by default, and the alternatives worth considering when that default does not fit.

When selecting a sensor for a wearable hand input device, five properties matter most:

- **Performance**: the sensor's features must be informative and low-noise, since the goal is to detect interactions as reliably as possible.
- **Power consumption**: wearable devices are only practical for everyday use if they can operate for a full day or longer without needing to be removed and recharged.
- **Form factor**: a sensing device worn on or near the hands must be lightweight and unobtrusive or users will not tolerate wearing it long-term.
- **Privacy**: users will not adopt a device if they feel it surveils them to an unnecessary or uncomfortable degree.
- **Calibration requirement**: a system that must be retrained for each user, or worse each time it is worn, cannot ship.

9.1 When Sensor Fusion Helps

Across most of my projects, a trend occurred: fusing more sensing modalities improved interaction detection. Sometimes the gain came from combining multiple instances of the same sensor, as in DiscoBand and Contextra, and sometimes from different sensors capturing aspects of an interaction that neither could capture

alone. In SmartPoser [59], adding UWB ranging to IMU data reduced median elbow error by 25.6% and wrist error by 20.4%, because UWB supplies the absolute measurement that inertial data lacks, helping to prevent drift. In Contextra [57], fusing multiple cameras with EMG and an IMU allowed more parts of a grasp to be sensed, improving precision by 0.16 and recall by 0.10 over EMG and IMU alone.

The important qualification, which my later work made clear, is that these gains diminish sharply once a sensor is well placed. In TrigRing, the full four-sensor configuration achieved an F1-score of 0.953 against 0.939 for the mini IR camera by itself, a difference of 2.5%. A new sensor should only be used if it covers a known failure mode or blind spot of the existing sensors, not if it only adds a nominally independent channel.

9.2 Best Single Sensor: A Low-Resolution Camera

Cameras were the dominant sensor across my projects, and I believe they remain the strongest general-purpose choice. Cameras perform well because visual information is dense and comparatively low-noise for hand interactions. Cameras also often require no calibration, and mirror how humans naturally perceive their hand interactions. Machine learning compounds these advantages, since deep learning models and vision-language models are exceptionally well optimized for visual features and benefit from a large corpus of training data no other modality can match. Audio is the nearest parallel, but the acoustic features produced by the hands are far less rich. IMU motion and vibration data have an even lower feature depth, and considerably more noise. Finally, bespoke sensors such as EMG or acoustic sonar can capture signals no other sensor can, but lack large amounts of training data and can require extensive per-user calibration.

For a single sensor, my recommendation is for a **low-resolution camera**. This can be achieved either with a camera that is natively low-resolution or with a higher-resolution camera whose output is downsampled, since both approaches reduce power draw and privacy exposure. The benefit is substantial; high-resolution cameras and small-package depth sensors can consume hundreds of milliwatts. In contrast, the natively low-resolution mini IR cameras used in Contextra [57] (30 × 30) and TrigRing (60 × 60) sensed complex interactions while consuming only ~10 mW each, roughly an order of magnitude less power. Industry trends also support this recommendation. Always-on camera modules operating below one milliwatt were commercially available as early as 2020 [107], and by 2026 major vendors were shipping detect-and-wake sensors in optical formats designed specifically for smart glasses [272]. I expect these trends to continue, making cameras an even more competitive option for wearable sensing.

The main weakness of camera-based sensing is privacy. Cameras capture rich visual data that can inadvertently include identifiable information about the user and, in many configurations, bystanders. This should be treated as a serious barrier to adoption, and not one designers can easily address after the fact. Some mitigations are possible, such as on-device inference with no image retention or

the active illumination technique I used in Contextra [57] and TrigRing, which removed the background of an image and left only the part near the hand. Nevertheless, privacy-preserving processing should be treated as a design requirement whenever a camera-based approach is chosen, not as merely an afterthought once the sensing pipeline is built.

9.3 Best Sensor Fusion: Multiple Cameras

A cross-cutting trend of my work was that sensor fusion generally improves performance. Given this, when a single sensor is not enough, my recommendation is to fuse information from multiple cameras, rather than pairing a camera with a different sensor type. One of the most significant limitations of cameras, especially with the small form factor required for wearables, is a small field of view, which creates problems with occlusion whenever an interaction falls behind a body part or just outside the frame. Optical fixes such as fisheye lens stacks tend to be prohibitively bulky, so the more practical alternative is to use several small cameras, each oriented to capture a different part of an interaction. I used this approach successfully in several of my projects (DiscoBand [58], Contextra [57]), and I believe it generalizes well. Multi-view camera sensing preserves the performance, power, form factor, and calibration advantages of cameras described above while directly addressing the field-of-view problem, at the cost of a proportional increase in total power draw and privacy exposure.

Multi-camera fusion is even more valuable for wearables further from the point of interaction. My results support this directly. TrigRing was able to use only one camera because its ring was worn directly on the finger. In contrast, in Contextra [57], no single wrist camera could see the entire grasp interaction, while combining all three improved precision by 0.22 and recall by 0.18 over any single camera alone. DiscoBand [58], also on the wrist, extended the same logic to sixteen mini depth cameras, where overlapping views recovered parts of the hand that any single view lost to self-occlusion.

9.4 Promising Alternatives for Constrained Contexts

Cameras are not always the right choice. Power budgets, form factor, or privacy requirements will sometimes rule them out, and in these cases I have two promising runner-up recommendations, each suited to a different set of constraints.

The first category includes sensors that measure the propagation of pressure waves: IMUs and microphones. These are the most widely available sensors in current wearables and for good reason. They are cheap, very low-power, drawing a few milliwatts or less, and produce simple one-dimensional signals. Their simplicity allows for processing to happen entirely on-device, which is faster and saves transmission power. They are also among the smallest and most mature sensor

packages available, and are already standard in commodity wearables. Privacy is strong as well, since an IMU cannot capture visually identifiable information and a microphone used only to detect a tap or vibration spike, rather than to record or transcribe speech, does not need to retain or interpret audio content. Finally, calibration demands are generally minimal, though drift in IMUs can require periodic recalibration. Looking to the future, MEMS technologies for IMUs and microphones are widely considered mature, though OEMs continue to realize incremental gains in cost, size, and performance [335]. Because these sensors already operate in the microwatt-to-milliwatt range (Table 8.1), further improvements yield diminishing absolute savings, especially compared to advances in cameras.

The second promising alternative is active acoustic sensing, distinct from the passive microphones just described. Active acoustic sensors emit inaudible ultrasonic chirps and measure their echoes. I explored this modality only in my work on TrigRing, but prior work has repeatedly shown it to be a high-performing sensor across many interaction types [157, 182, 338, 339]. Its main advantage is the ability to sense subtle changes in geometry over a large area, including depth information that many cheap cameras struggle to capture. This makes active acoustic sensing especially compelling when depth is the primary signal of interest. It provides richer depth information than 1D laser rangefinders, at lower power than miniature depth cameras. Like IMUs and microphones, active acoustic sensors do not capture visually identifiable information, making them a comparatively privacy-preserving option for depth-sensitive applications where a camera would otherwise be the default choice. Their main limitation is power. For my acoustic sensor in TrigRing, amplifying the chirp signal and driving the transducer accounted for 85% of the power consumption. Unlike image sensors, which have benefited directly from CMOS process scaling, analog and mixed-signal blocks of this kind have historically gained far less from newer process nodes [7]. As such, I expect this figure to improve more slowly than camera power has in the future. Active acoustic sensing also typically requires some environment-specific tuning, such as accounting for ambient acoustic reflections, which is a modest but non-trivial calibration burden relative to cameras or IMUs.

9.5 Toward Proactive, Context-Aware Devices

More than a ranking of individual sensors, these recommendations point toward sensing modalities which can support the kind of devices this thesis argues we need. As I discuss throughout this work, the rise of AI-driven mobile applications exposes the limitations of input frameworks that are reactive and lack context about the user's physical situation. Despite their power and privacy costs, cameras provide the most direct path to sensing rich, continuous context that proactive, AI-driven systems need to reason about what a user needs next. At present, clever tricks such as the wake trigger explored in TrigRing can get around some of the power and privacy limitations. In the future, industry trends point to improvements in camera sensor dies and on-device compute that I believe could address

these limitations further and lead to cameras and camera fusions becoming the dominant sensors for wearable, context-aware devices.

Chapter 10

Conclusion

Throughout my research, I have focused on how wearable devices can sense implicit user behaviors to address the limitations of traditional input methods. As illustrated in Figure 1.1, this work contributes to a vision of expanding the input capabilities of wearable devices beyond what is currently possible while preserving the mobility and convenience that make these devices valuable. I pursue this vision through three complementary strengths. First, I draw on expertise in human factors to understand users and the challenges they encounter in everyday interactions. Second, I leverage a deep understanding of hardware design to develop novel sensors and sensing techniques that capture how people move, behave, and interact with the world around them. Third, I apply methods from artificial intelligence and machine learning to interpret these signals, transforming raw sensor data into meaningful models of user behavior. Together, these strengths have enabled the development of new forms of wearable input that address longstanding limitations in existing systems. To advance this vision, I have explored both novel sensing hardware and innovative uses of existing sensors, with the goal of creating input experiences that are more capable, practical, and robust.

My work approaches the wearable input problem from multiple perspectives, exploring techniques that improve a device's implicit awareness of user behavior at different levels of abstraction. At the most direct level, this includes support for explicit input through the tracking of hand gestures [58] and touch interactions [60, 202]. However, it also extends to the sensing of more indirect behaviors that occur in the background, including touch input [60, 202], body pose [4, 58, 59], and object interactions [57]. These implicit behaviors do not function as input by themselves; instead, they provide rich contextual information that can support an AI-driven input layer. By automatically inferring a user's context, future wearable systems can better anticipate needs and intentions, reducing reliance on explicit interaction. Collectively, this research demonstrates how wearable devices can become more proactive, contextually aware, and ultimately easier to use by narrowing the gap between what users do and what devices are truly capable of.

10.1 Thesis Contributions

This thesis contributes a new direction for interaction with wearable and mobile devices. Historically, input on wearable devices has been constrained by their small form factors and reliance on traditional input mechanisms such as buttons and touchscreens. As devices have become smaller and more mobile, the available input bandwidth has often decreased, creating an input gap between what users wish to communicate and what devices can effectively capture. Rather than treating this limitation as an unavoidable consequence of miniaturization, my work demonstrates how wearable devices can leverage their unique positions on the body to sense naturally occurring hand interactions in the surrounding environment. These sensed interactions can support new forms of explicit input, but more importantly, they enable the capture of implicit behavioral signals that provide rich contextual information about the user.

This emphasis on implicit sensing represents the broader contribution of this thesis. While the proposed systems help address the existing input gap, they also point toward a new interaction paradigm in which devices rely less on direct commands and more on an understanding of user context. In our time, advances in artificial intelligence have dramatically increased the capability of digital systems, often surpassing the rate at which users can provide manual input. In digital environments, this challenge has been addressed through agent-based systems that can act on a user's behalf after receiving high-level instructions. However, an equivalent approach for the physical world remains largely unexplored. Because human activities and needs are fundamentally grounded in the physical world, enabling devices to understand physical context is a critical step toward more intelligent and helpful computing systems.

The sensing approaches developed throughout this thesis provide a foundation for bridging this gap. By capturing implicit hand interactions and other behavioral signals, wearable devices can gain awareness of what users are doing, interacting with, and potentially intending to do next. This contextual understanding can augment existing AI assistants, enabling them to provide assistance that is informed not only by digital information but also by the user's physical environment and ongoing activities. Whether future intelligent systems take the form of wearable assistants, ambient computing platforms, or personal robots, I believe human-centered interaction will remain essential. As intelligent systems continue to evolve, users will increasingly communicate through behavior and context rather than explicit commands alone. This thesis argues that implicit sensing of everyday interactions provides a practical path toward that future.

10.2 Future Directions

10.2.1 General-Purpose Implicit Input

While I have made significant progress, there is still work to be done to realize the full vision of automated contextual assistants on consumer devices. One area I am interested in exploring in the future is broadening the scope of behaviors a system can recognize. To achieve their full value in helping users, agentic systems need to be able to recognize a broad scope of different behaviors. This is because users do a lot of different things. Take a system which only works in a "narrow" context like helping a woodworker with their craft. You might want to recognize things like taking a measurement, performing a glue-up, sanding a piece of wood, and brushing on a varnish. But even in a woodshop, there are still an overwhelming number of different motions, objects, and activities a system might want to recognize.

Considering this, one direction for my future research is to work towards general-purpose implicit input. My work on implicit input so far has mostly been fixed at intermediate outputs before the application layer like touch, pose, and grasp. My desire is to level-up to an input layer that can automatically recognize all tasks and behaviors of the hands. In the old reactive input framework, interaction was event-driven with each action translating to a change in a GUI or application. In my new proactive framework, there is no fixed API of events. Instead, the input system will recognize and supply a summary of the current context that an AI layer can translate into actions for the end application. One of my hopes for this direction is to broaden the validity of my work by being able to conduct studies not just in lab environments with fixed procedures, but rather out in the world as users perform their natural daily activities.

10.2.2 Continuous Implicit Input

A primary method I foresee enabling general-purpose input is combining hand sensing with powerful multi-modal models (e.g., VLMs) from egocentric cameras to recognize broad classes of behaviors. While this approach, particularly with the recent incorporation of foundation models, has shown great promise for recognizing a wide array of activities with high accuracy, it is also not scalable for the future of implicit input. This is because the high-res cameras required consume a significant amount of power. The use of such cameras on a mobile device, combined with the power needed for processing and compute, can shorten battery life to an unusably low amount. In this way, power consumption prevents current egocentric camera systems from being used for all-day implicit input tracking.

Thus, another direction I plan to pursue is working towards continuous implicit input systems that can run all day on a battery. The whole idea of implicit input only works if it can run uninterrupted throughout the day. If users must manually choose when their devices are helping, systems are reverting back to explicit input. However, currently it is infeasible to continuously run agents that

generate their context through camera-based scene understanding. Not only is the power cost of compute too high, but at a hardware level even the camera sensors cannot run so long on a wearable-sized battery.

In my latest projects on grasp detection, Contextra and TrigRing, I began to explore this problem through the lens of device constellations using tiered sensing to work together. While I believe there are still interesting questions to explore with tiered sensing, I am also curious about investigating alternative approaches. Some directions I am interested in include creating intelligent wake schedules for VLMs based on past behavior, or having devices communicate with smart, instrumented environments to offload some of the continuous sensing burden from battery-bound devices.

10.2.3 Timely Implicit Input

The final area I plan to explore is the development of more timely forms of implicit input. Throughout this thesis, I have argued for a shift from purely reactive devices toward systems that can proactively support users based on contextual understanding. However, accurately sensing user context is only part of the challenge. Equally important is determining when assistance should be provided and how it should be delivered. Future work should investigate questions such as: What is the appropriate balance between utility and interruption? How can systems identify moments when assistance is likely to be beneficial without becoming distracting, intrusive, or overwhelming?

As computing systems become increasingly capable of understanding users and their environments, these questions will be critical to realizing a future of ubiquitous intelligent assistance. While advances in sensing can enable devices to infer what users are doing, designing systems that understand when users want assistance, and how that assistance should be presented, remains a largely open problem. Compared to the other research directions discussed in this thesis, this challenge is broader and less clearly defined. Nevertheless, I believe it will be essential for ensuring that future forms of implicit input are not only technically capable, but also genuinely useful, acceptable, and desirable to use in everyday life.

Bibliography

[1] Alireza Abedin, Mahsa Ehsanpour, Qinfeng Shi, Hamid Rezatofighi, and Damith C. Ranasinghe. 2021. Attend and Discriminate: Beyond the State-of-the-Art for Human Activity Recognition Using Wearable Sensors. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 1 (March 2021), 1:1–1:22. DOI:http://dx.doi.org/10.1145/3448083

[2] Y. Adajania, J. Gosalia, A. Kanade, H. Mehta, and N. Shekokar. 2010. Virtual Keyboard Using Shadow Analysis. In *2010 3rd International Conference on Emerging Trends in Engineering and Technology*. 163–165. DOI:http://dx.doi.org/10.1109/ICETET.2010.115 ISSN: 2157-0485.

[3] Ankur Agarwal, Shahram Izadi, Manmohan Chandraker, and Andrew Blake. 2007. High Precision Multi-touch Sensing on Surfaces using Overhead Cameras. *Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'07)* (Oct. 2007), 197–200. DOI:http://dx.doi.org/10.1109/TABLETOP.2007.29 Conference Name: Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'07) ISBN: 9780769530130 9780769520131 Place: Newport, RI, USA Publisher: IEEE.

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

[5] Last Reviewed: American Heart Association. 2024. Medication Adherence: Taking Your Meds as Directed. (Jan. 2024). https://www.heart.org/en/health-topics/consumer-healthcare/medication-information/medication-adherence-taking-your-meds-as-directed

[6] Farshid Amirabdollahian and Michael Walters. 2017. Application of support vector machines to detect hand and wrist gestures using a myoelectric armband. (July 2017). http://uhra.herts.ac.uk/handle/2299/19095 Accepted: 2017-07-26T10:22:36Z.

[7] Anne-Johan Annema, Bram Nauta, Ronald van Langevelde, and Hans Tuinhout. 2005. Analog Circuits in Ultra-Deep-Submicron CMOS. *IEEE Journal of Solid-State Circuits* 40, 1 (2005), 132–143. DOI:http://dx.doi.org/10.1109/JSSC.2004.837247

[8] Apple. 2020. AirTag. (2020). https://www.apple.com/airtag/

[9] Apple. 2022a. Core Motion | Apple Developer Documentation. (2022). https://developer.apple.com/documentation/coremotion/

[10] Apple. 2022b. Nearby Interaction | Apple Developer Documentation. (2022). https://developer.apple.com/documentation/nearbyinteraction

[11] Apple. 2023. Set up Handwashing on Apple Watch. (2023). https://support.apple.com/guide/watch/set-up-handwashing-apdc9b9f04a8/watchos

[12] Apple. 2024a. Apple Vision Pro. (2024). https://www.apple.com/apple-vision-pro/

[13] Apple. 2024b. Connect and charge Apple Vision Pro battery. (2024). https://support.apple.com/en-us/117740

[14] 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*. ACM, Virtual Event Canada, 1–10. DOI:http://dx.doi.org/10.1145/3385959.3418450

[15] Arducam. 2023a. Arducam 12MP IMX708 Fixed Focus HDR High SNR Camera Module for Raspberry Pi. (2023). https://www.arducam.com/product/arducam-12mp-imx708-fixed-focus-hdr-high-snr-camera-module-for-raspberry-pi/

[16] Arducam. 2023b. Arducam OV2640 Camera Module, 2MP Mini CCM Compact Camera Modules Compatible with Arduino ESP32 ESP8266 Development Board with DVP 24 Pin Interface. (2023). https://www.arducam.com/product/arducam-ov2640-camera-module-2mp-mini-ccm-compact-camera-modules-compatible-with-arduino_m0031esp32-esp8266-development-board-with-dvp-24-pin-interface_/

[17] Arducam. 2023c. Arducam OV7670 Camera Module, VGA Mini CCM Compact Camera Modules Compatible with Arduino ARM FPGA, with DVP 24 Pin Interface. (2023). https://www.arducam.com/product/arducam_ov7670_camera_module_vga_mini_ccm_compact_camera_modules_m0030/

[18] John Arevalo, Thamar Solorio, Manuel Montes-y Gómez, and Fabio A. González. 2017. Gated Multimodal Units for Information Fusion. (Feb. 2017). DOI:http://dx.doi.org/10.48550/arXiv.1702.01992 arXiv:1702.01992 [stat.ML].

[19] Eric R. Bachmann, Robert B. McGhee, Xiaoping Yun, and Michael J. Zyda. 2001. Inertial and magnetic posture tracking for inserting humans into networked virtual environments. In *Proceedings of the ACM symposium on Virtual reality software and technology (VRST '01)*. Association for Computing Machinery, New York, NY, USA, 9–16. DOI:http://dx.doi.org/10.1145/505008.505011

[20] Seungyeol Baek, Yoonbyung Chai, Yonghyeon Lee, Sungjoon Choi, and Sungho Suh. 2026. Rotation-Invariant Multi-IMU Activity Recognition under Independent Per-Location Orientation Shifts. (Aug. 2026). DOI:http://dx.doi.org/10.1145/3830727.3834824 arXiv:2608.15621 [cs.AI].

[21] Alejandro Baldominos, Yago Saez, and Cristina García del Pozo. 2015. An Approach to Physical Rehabilitation Using State-of-the-art Virtual Reality and Motion Tracking Technologies. *Procedia Computer Science* 64 (2015), 10–16. DOI:http://dx.doi.org/10.1016/j.procs.2015.08.457

[22] Sven Bambach, Stefan Lee, David J. Crandall, and Chen Yu. 2015. Lending A Hand: Detecting Hands and Recognizing Activities in Complex Egocentric Interactions. In *2015 IEEE International Conference on Computer Vision (ICCV)*. 1949–1957. DOI:http://dx.doi.org/10.1109/ICCV.2015.226 ISSN: 2380-7504.

[23] Ling Bao and Stephen S. Intille. 2004. Activity Recognition from User-Annotated Acceleration Data. In *Pervasive Computing (Lecture Notes in Computer Science)*, Alois Ferscha and Friedemann Mattern (Eds.). Springer, Berlin, Heidelberg, 1–17. DOI:http://dx.doi.org/10.1007/978-3-540-24646-6_1

[24] Gerald Baulig, Thomas Gulde, and Cristóbal Curio. 2018. Adapting Egocentric Visual Hand Pose Estimation Towards a Robot-Controlled Exoskeleton. In *Computer Vision – ECCV 2018 Workshops: Munich, Germany, September 8-14, 2018, Proceedings, Part VI*. Springer-Verlag, Berlin, Heidelberg, 241–256. DOI:http://dx.doi.org/10.1007/978-3-030-11024-6_16

[25] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann. 2020. BlazePose: On-device Real-time Body Pose tracking. (2020). DOI:http://dx.doi.org/10.48550/ARXIV.2006.10204

[26] Abdelkareem Bedri, Diana Li, Rushil Khurana, Kunal Bhuwalka, and Mayank Goel. 2020. FitByte: Automatic Diet Monitoring in Unconstrained Situations Using Multimodal Sensing on Eyeglasses. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. DOI: http://dx.doi.org/10.1145/3313831.3376869

[27] Abdelkareem Bedri, Richard Li, Malcolm Haynes, Raj Prateek Kosaraju, Ishaan Grover, Temiloluwa Prioleau, Min Yan Beh, Mayank Goel, Thad Starner, and Gregory Abowd. 2017. EarBit: Using Wearable Sensors to Detect Eating Episodes in Unconstrained Environments. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 3, Article 37 (Sept. 2017), 20 pages. DOI:http://dx.doi.org/10.1145/3130902

[28] Hrvoje Benko and Andrew D. Wilson. 2009. DepthTouch: Using Depth-Sensing Camera to Enable Freehand Interactions On and Above the Interactive Surface.

[29] Eugen Berlin, Jun Liu, Kristof van Laerhoven, and Bernt Schiele. 2010. Coming to grips with the objects we grasp: detecting interactions with efficient wrist-worn sensors. In *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction (TEI '10)*. Association for Computing Machinery, New York, NY, USA, 57–64. DOI:http://dx.doi.org/10.1145/1709886.1709898

[30] Yannick Bernaerts, Jo Vermeulen, Matthias Druwé, Johannes Schöning, and Sebastiaan Steensels. 2014. The Office Smartwatch – Development and Design of a Smartwatch App to Digitally Augment Interactions in an Office Environment. (2014), 4.

[31] Sarnab Bhattacharya, Rebecca Adaimi, and Edison Thomaz. 2022. Leveraging Sound and Wrist Motion to Detect Activities of Daily Living with Commodity Smartwatches. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 2 (July 2022), 42:1–42:28. DOI:http://dx.doi.org/10.1145/3534582

[32] Ali Bigdelou, Loren Schwarz, and Nassir Navab. 2012. An Adaptive Solution for Intra-Operative Gesture-Based Human-Machine Interaction. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces (IUI '12)*. Association for Computing Machinery, New York, NY, USA, 75–84. DOI:http://dx.doi.org/10.1145/2166966.2166981

[33] Richard A. Bolt. 1980. Put-that-there: Voice and gesture at the graphics interface. *ACM SIGGRAPH Computer Graphics* 14, 3 (July 1980), 262–270. DOI:http://dx.doi.org/10.1145/965105.807503

[34] Pietro Bonazzi, Julian Moosmann, Ahmet Celik, Philipp Mayer, and Michele Magno. 2026. OpenGlass: Ultra-Low-Power On-Device AI Eyewear with Event-based Vision. (June 2026). DOI:http://dx.doi.org/10.48550/arXiv.2606.07431 arXiv:2606.07431 [cs.CV].

[35] Frédéric Bosché, Mohamed Abdel-Wahab, and Ludovico Carozza. 2016. Towards a Mixed Reality System for Construction Trade Training. *Journal of Computing in Civil Engineering* 30, 2 (March 2016), 04015016. DOI:http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000479 Publisher: American Society of Civil Engineers.

[36] Broadcom. 2016. APDS-9500 | Imaging Gesture and Proximity Sensor. (2016). https://www.broadcom.com/products/optical-sensors/proximity-sensors/apds-9500

[37] Thisum Buddhika, Haimo Zhang, Chamod Weerasinghe, Suranga Nanayakkara, and Roger Zimmermann. 2019. OSense: Object-activity Identification Based on Gasping Posture and Motion. In *Proceedings of the 10th Augmented Human International Conference 2019 (AH2019)*. Association for Computing Machinery, New York, NY, USA, 1–5. DOI:http://dx.doi.org/10.1145/3311823.3311841

[38] Hammad Tanveer Butt, Bertram Taetz, Mathias Musahl, Maria A. Sanchez, Pramod Murthy, and Didier Stricker. 2021. Magnetometer Robust Deep Human Pose Regression With Uncertainty Prediction Using Sparse Body Worn Magnetic Inertial Measurement Units. *IEEE Access* 9 (2021), 36657–36673. DOI:http://dx.doi.org/10.1109/ACCESS.2021.3062545 Conference Name: IEEE Access.

[39] Minjie Cai, Kris M. Kitani, and Yoichi Sato. 2017. An Ego-Vision System for Hand Grasp Analysis. *IEEE Transactions on Human-Machine Systems* 47, 4 (Aug. 2017), 524–535. DOI: http://dx.doi.org/10.1109/THMS.2017.2681423

[40] Minjie Cai, Kris M. Kitani, and Yoichi Sato. 2018. Understanding hand-object manipulation by modeling the contextual relationship between actions, grasp types and object attributes. *ArXiv* (July 2018). https://www.semanticscholar.org/paper/Understanding-hand-object-manipulation-by-modeling-Cai-Kitani/9c1b526dc9adba029809e45693f758c2af913f60

[41] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2019. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *arXiv:1812.08008 [cs]* (May 2019). http://arxiv.org/abs/1812.08008 arXiv: 1812.08008.

[42] Maria Claudia F Castro. 2015. Selection of suitable hand gestures for reliable myoelectric human computer interface. (2015), 11.

[43] Liwei Chan, Yi-Ling Chen, Chi-Hao Hsieh, Rong-Hao Liang, and Bing-Yu Chen. 2015. CyclopsRing: Enabling Whole-Hand and Context-Aware Interactions Through a Fisheye Ring. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, Charlotte NC USA, 549–556. DOI:http://dx.doi.org/10.1145/2807442.2807450

[44] Dibyadip Chatterjee, Fadime Sener, Shugao Ma, and Angela Yao. 2023. Opening the Vocabulary of Egocentric Actions. (2023). https://arxiv.org/abs/2308.11488

[45] Baiyu Chen, Wilson Wongso, Zechen Li, Yonchanok Khaokaew, Hao Xue, and Flora D. Salim. 2026. COMODO: Cross-Modal Video-to-IMU Distillation for Efficient Egocentric Human Activity Recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 10, 2 (June 2026), 33:1–33:29. DOI:http://dx.doi.org/10.1145/3810218

[46] Feiyu Chen, Jia Deng, Zhibo Pang, Majid Baghaei Nejad, Huayong Yang, and Geng Yang. 2018. Finger Angle-Based Hand Gesture Recognition for Smart Infrastructure Using Wearable Wrist-Worn Camera. *Applied Sciences* 8, 3 (2018). DOI:http://dx.doi.org/10.3390/app8030369

[47] Nutan Chen, Göran Westling, Benoni B. Edin, and Patrick Van Der Smagt. 2020b. Estimating Fingertip Forces, Torques, and Local Curvatures from Fingernail Images. *Robotica* 38, 7 (July 2020), 1242–1262. DOI:http://dx.doi.org/10.1017/S0263574719001383

[48] Tuochao Chen, Benjamin Steeper, Kinan Alsheikh, Songyun Tao, François Guimbretière, and Cheng Zhang. 2020a. C-Face: Continuously Reconstructing Facial Expressions by Deep Learning Contours of the Face with Ear-mounted Miniature Cameras. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20)*. Association for Computing Machinery, New York, NY, USA, 112–125. DOI:http://dx.doi.org/10.1145/3379337.3415879

[49] Pascal Chiu, Kazuki Takashima, Kazuyuki Fujita, and Yoshifumi Kitamura. 2019. Pursuit Sensing: Extending Hand Tracking Space in Mobile VR Applications. (2019), 5.

[50] Savannah Cofer, Tyler N. Chen, Jackie Junrui Yang, and Sean Follmer. 2022. Detecting Touch and Grasp Gestures Using a Wrist-Worn Optical and Inertial Sensing Network. *IEEE Robotics and Automation Letters* 7, 4 (Oct. 2022), 10842–10849. DOI:http://dx.doi.org/10.1109/LRA.2022.3191173

[51] J. A. Corrales, F. A. Candelas, and F. Torres. 2008. Hybrid tracking of human operators using IMU/UWB data fusion by a Kalman filter. In *2008 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 193–200. DOI:http://dx.doi.org/10.1145/1349822.1349848 ISSN: 2167-2148.

[52] Guangzhao Dai, Xiangbo Shu, Wenhao Wu, Rui Yan, and Jiachao Zhang. 2024. GPT4Ego: Unleashing the Potential of Pre-trained Models for Zero-Shot Egocentric Action Recognition. (May 2024). DOI:http://dx.doi.org/10.48550/arXiv.2401.10039 arXiv:2401.10039 [cs.CV].

[53] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. 2020. The EPIC-KITCHENS Dataset: Collection, Challenges and Baselines. (April 2020). DOI:http://dx.doi.org/10.48550/arXiv.2005.00343 arXiv:2005.00343 [cs.CV].

[54] T DeFanti and DJ Sandin. 1977. Sayre Glove Final Project Report. *US NEA R60-34-163 Final Project Report* (1977).

[55] Artem Dementyev and Joseph A. Paradiso. 2014. WristFlex: low-power gesture input with wrist-worn pressure sensors. In *Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST '14)*. Association for Computing Machinery, New York, NY, USA, 161–166. DOI:http://dx.doi.org/10.1145/2642918.2647396

[56] Shrivatsa Deshmukh, Vitthal Khatik, and Anupam Saxena. 2023. Robust Fusion Model for Handling EMG and Computer Vision Data in Prosthetic Hand Control. *IEEE Sensors Letters* 7, 9 (Sept. 2023), 1–4. DOI:http://dx.doi.org/10.1109/LSENS.2023.3301837 Conference Name: IEEE Sensors Letters.

[57] Nathan DeVrio, Roger Boldu, Eric Whitmire, and Wolf Kienzle. 2025. Contextra: Detecting Object Grasps With Low-Power Cameras and Sensor Fusion On the Wrist. *Proc. ACM Hum.-Comput. Interact.* 9, 5, Article MHCI006 (Sept. 2025), Article MHCI006, 25 pages. DOI:http://dx.doi.org/10.1145/3743741

[58] Nathan Devrio and Chris Harrison. 2022. DiscoBand: Multiview Depth-Sensing Smartwatch Strap for Hand, Body and Environment Tracking. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology (UIST '22)*. Association for Computing Machinery, New York, NY, USA, 1–13. DOI:http://dx.doi.org/10.1145/3526113.3545634

[59] Nathan DeVrio, Vimal Mollyn, and Chris Harrison. 2023. SmartPoser: Arm Pose Estimation with a Smartphone and Smartwatch Using UWB and IMU Data. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, NY, USA. DOI:http://dx.doi.org/10.1145/3586183.3606821

[60] Nathan DeVrio, Vimal Mollyn, and Chris Harrison. 2026. VelociTrack: Touch Input On Uninstrumented Surfaces Using High-Speed Headset Cameras. *Proc. ACM Hum.-Comput. Interact.* 10, 5, Article MHCI006 (Sept. 2026), Article MHCI006, 25 pages. DOI:http://dx.doi.org/10.1145/3821693

[61] Han Ding, Jinsong Han, Longfei Shangguan, Wei Xi, Zhiping Jiang, Zheng Yang, Zimu Zhou, Panlong Yang, and Jizhong Zhao. 2017. A Platform for Free-Weight Exercise Monitoring with Passive Tags. *IEEE Transactions on Mobile Computing* 16, 12 (Dec. 2017), 3279–3293. DOI:http://dx.doi.org/10.1109/TMC.2017.2691705 Conference Name: IEEE Transactions on Mobile Computing.

[62] Yujie Dong, Adam Hoover, Jenna Scisco, and Eric Muth. 2015. A New Method for Measuring Meal Intake in Humans via Automated Wrist Motion Tracking. (2015), 25.

[63] Camille Dupré, Caroline Appert, Stéphanie Rey, Houssem Saidi, and Emmanuel Pietriga. 2024. TriPad: Touch Input in AR on Ordinary Surfaces with Hand Tracking Only. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, Honolulu HI USA, 1–18. DOI:http://dx.doi.org/10.1145/3613904.3642323

[64] Hugh Durrant-Whyte and Tim Bailey. 2006. Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms. (2006), 9.

[65] Umm e Sadima, Yazeed Alkhrijah, Danish Hamid, Muhammad Ehatisham Ul Haq, Syed Muhammad Usman, Shehzad Khalid, and Mohamad A. Alawad. 2025. EgoVision a YOLO-ViT hybrid for robust egocentric object recognition. *Scientific Reports* 15, 1 (Oct. 2025), 34723. DOI:http://dx.doi.org/10.1038/s41598-025-18341-y

[66] Ethan Eddy, Erik J Scheme, and Scott Bateman. 2023. A Framework and Call to Action for the Future Development of EMG-Based Input in HCI. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–23. DOI:http://dx.doi.org/10.1145/3544548.3580962

[67] M. El-Gohary and J. McNames. 2012. Shoulder and Elbow Joint Angle Tracking With Inertial Sensors. *IEEE Transactions on Biomedical Engineering* 59, 9 (Sept. 2012), 2635–2641. DOI:http://dx.doi.org/10.1109/TBME.2012.2208750 Conference Name: IEEE Transactions on Biomedical Engineering.

[68] ELP. 2020. ELP High Speed USB3.0 USB Camera 2MP USB Camera Module with IMX291 Image Sensor, 1920 * 1080@50fps Webcam with 100 Degree No Distortion Lens for Android Windows Linux,Plug&Play Webcam [ELP-SUSB1080P01-LC1100] - $60.90 : ELP USB Webcam. (2020). http://www.webcamerausb.com/elp-high-speed-usb30-usb-camera-2mp-usb-camera-module-with-imx291-image-sensor-1920-1080 50f html

[69] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*. AAAI Press, 226–231.

[70] Junjun Fan, Xiangmin Fan, Feng Tian, Yang Li, Zitao Liu, Wei Sun, and Hongan Wang. 2018. What is That in Your Hand?: Recognizing Grasped Objects via Forearm Electromyography Sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 4 (Dec. 2018), 1–24. DOI:http://dx.doi.org/10.1145/3287039

[71] Neil Xu Fan and Robert Xiao. 2022. Reducing the Latency of Touch Tracking on Ad-hoc Surfaces. *Proc. ACM Hum.-Comput. Interact.* 6, ISS (Nov. 2022), 577:489–577:499. DOI:http://dx.doi.org/10.1145/3567730

[72] Biyi Fang, Nicholas D. Lane, Mi Zhang, Aidan Boran, and Fahim Kawsar. 2016. BodyScan: Enabling Radio-based Sensing on Wearable Devices for Contactless Activity and Vital Sign Monitoring. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '16)*. Association for Computing Machinery, New York, NY, USA, 97–110. DOI:http://dx.doi.org/10.1145/2906388.2906411

[73] Yazan Abu Farha and Juergen Gall. 2019. MS-TCN: Multi-Stage Temporal Convolutional Network for Action Segmentation. (April 2019). DOI:http://dx.doi.org/10.48550/arXiv.1903.01945 arXiv:1903.01945 [cs.CV].

[74] A. Feldman, E.M. Tapia, S. Sadi, P. Maes, and C. Schmandt. 2005. ReachMedia: On-the-move interaction with everyday objects. In *Ninth IEEE International Symposium on Wearable Computers (ISWC'05)*. IEEE, Osaka, Japan, 52–59. DOI:http://dx.doi.org/10.1109/ISWC.2005.44

[75] K.P. Fishkin, M. Philipose, and A. Rea. 2005. Hands-On RFID: Wireless Wearables for Detecting Use of Objects. In *Ninth IEEE International Symposium on Wearable Computers (ISWC'05)*. IEEE, Osaka, Japan, 38–43. DOI:http://dx.doi.org/10.1109/ISWC.2005.25

[76] Ray-Ban Meta Forum. 2026. (11) Ray-ban meta Forum | One question ♂, yesterday I recorded 10 videos of 3 minutes each with my Ray-Ban Meta 1st Gen, and the battery went from 100% down to ... | Facebook. (2026). https://www.facebook.com/groups/395416269682034/posts/851688027388187/

[77] Markus Funk, Stefan Schneegass, Michael Behringer, Niels Henze, and Albrecht Schmidt. 2015. An Interactive Curtain for Media Usage in the Shower. In *Proceedings of the 4th International Symposium on Pervasive Displays*. ACM, Saarbruecken Germany, 225–231. DOI:http://dx.doi.org/10.1145/2757710.2757713

[78] F. Gemperle, C. Kasabach, J. Stivoric, M. Bauer, and R. Martin. 1998. Design for wearability. In *Digest of Papers. Second International Symposium on Wearable Computers (Cat. No.98EX215)*. 116–122. DOI:http://dx.doi.org/10.1109/ISWC.1998.729537

[79] Jun Gong, Aakar Gupta, and Hrvoje Benko. 2020. Acustico: Surface Tap Detection and Localization using Wrist-based Acoustic TDOA Sensing. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20)*. Association for Computing Machinery, New York, NY, USA, 406–419. DOI:http://dx.doi.org/10.1145/3379337.3415901

[80] Jun Gong, Xing-Dong Yang, and Pourang Irani. 2016. WristWhirl: One-Handed Continuous Smartwatch Input Using Wrist Gestures. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (2016) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 861–872. DOI:http://dx.doi.org/10.1145/2984511.2984563

[81] Jun Gong, Yang Zhang, Xia Zhou, and Xing-Dong Yang. 2017. Pyro: Thumb-Tip Gesture Recognition Using Pyroelectric Infrared Sensing. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. ACM, Québec City QC Canada, 553–563. DOI:http://dx.doi.org/10.1145/3126594.3126615

[82] Google. 2020a. MediaPipe Hands. (2020). https://google.github.io/mediapipe/solutions/hands.html

[83] Google. 2020b. MediaPipe Pose. (2020). https://google.github.io/mediapipe/solutions/pose.html

[84] Google. 2022. Android Developers, Ultra-wideband (UWB) communication. (2022). https://developer.android.com/guide/topics/connectivity/uwb

[85] Patrick Grady, Jeremy A Collins, Chengcheng Tang, Christopher D Twigg, Kunal Aneja, James Hays, and Charles C Kemp. 2024. PressureVision++: Estimating Fingertip Pressure from Diverse RGB Images. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)* (2024).

[86] Patrick Grady, Chengcheng Tang, Samarth Brahmbhatt, Christopher D. Twigg, Chengde Wan, James Hays, and Charles C. Kemp. 2022. PressureVision: Estimating Hand Pressure from a Single RGB Image. (Sept. 2022). DOI:http://dx.doi.org/10.48550/arXiv.2203.10385 arXiv:2203.10385 [cs].

[87] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina Gonzalez, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jachym Kolar, Satwik Kottur,

Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbelaez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. 2022. Ego4D: Around the World in 3,000 Hours of Egocentric Video. (March 2022). DOI: http://dx.doi.org/10.48550/arXiv.2110.07058 arXiv:2110.07058 [cs.CV].

[88] Bernhard Großwindhager, Carlo Alberto Boano, Michael Rath, and Kay Römer. 2018. Enabling Runtime Adaptation of Physical Layer Settings for Dependable UWB Communications. In *2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*. 01–11. DOI:http://dx.doi.org/10.1109/WoWMoM.2018.8449776

[89] Yizheng Gu, Chun Yu, Zhipeng Li, Weiqi Li, Shuchang Xu, Xiaoying Wei, and Yuanchun Shi. 2019. Accurate and Low-Latency Sensing of Touch Contact on Any Surface with Finger-Worn IMU Sensor. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New York, NY, USA, 1059–1070. DOI:http://dx.doi.org/10.1145/3332165.3347947

[90] Yizheng Gu, Chun Yu, Zhipeng Li, Zhaoheng Li, Xiaoying Wei, and Yuanchun Shi. 2020. QwertyRing: Text Entry on Physical Surfaces Using a Ring. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4 (Dec. 2020), 128:1–128:29. DOI:http://dx.doi.org/10.1145/3432204

[91] Sean Gustafson, Christian Holz, and Patrick Baudisch. 2011. Imaginary phone: learning imaginary interfaces by transferring spatial memory from a familiar device. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, Santa Barbara California USA, 283–292. DOI:http://dx.doi.org/10.1145/2047196.2047233

[92] Nils Y. Hammerla, Shane Halloran, and Thomas Plötz. 2016. Deep, convolutional, and recurrent models for human activity recognition using wearables. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press, New York, New York, USA, 1533–1540.

[93] Shangchen Han, Beibei Liu, Randi Cabezas, Christopher D. Twigg, Peizhao Zhang, Jeff Petkau, Tsz-Ho Yu, Chun-Jung Tai, Muzaffer Akbay, Zheng Wang, Asaf Nitzan, Gang Dong, Yuting Ye, Lingling Tao, Chengde Wan, and Robert Wang. 2020. MEgATrack: monochrome egocentric articulated hand-tracking for virtual reality. *ACM Transactions on Graphics (TOG)* 39, 4 (Aug. 2020), 87:87:1–87:87:13. DOI:http://dx.doi.org/10.1145/3386569.3392452

[94] Shangchen Han, Po-chen Wu, Yubo Zhang, Beibei Liu, Linguang Zhang, Zheng Wang, Weiguang Si, Peizhao Zhang, Yujun Cai, Tomas Hodan, Randi Cabezas, Luan Tran, Muzaffer Akbay, Tsz-Ho Yu, Cem Keskin, and Robert Wang. 2022. UmeTrack: Unified multi-view end-to-end hand tracking for VR. In *SIGGRAPH Asia 2022 Conference Papers*. 1–9. DOI: http://dx.doi.org/10.1145/3550469.3555378 arXiv:2211.00099 [cs].

[95] Teng Han, Khalad Hasan, Keisuke Nakamura, Randy Gomez, and Pourang Irani. 2017. SoundCraft: Enabling Spatial Interactions on Smartwatches using Hand Generated Acoustics. (2017), 13.

[96] Harish Haresamudram, Irfan Essa, and Thomas Plötz. 2021. Contrastive Predictive Coding for Human Activity Recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 2 (June 2021), 65:1–65:26. DOI:http://dx.doi.org/10.1145/3463506

[97] Chris Harrison, Hrvoje Benko, and Andrew D. Wilson. 2011. OmniTouch: wearable multitouch interaction everywhere. In *Proceedings of the 24th annual ACM symposium on User interface software and technology (UIST '11)*. Association for Computing Machinery, New York, NY, USA, 441–450. DOI:http://dx.doi.org/10.1145/2047196.2047255

[98] Chris Harrison and Scott E. Hudson. 2008a. Lightweight material detection for placement-aware mobile computing. In *Proceedings of the 21st annual ACM symposium on User interface software and technology (UIST '08)*. Association for Computing Machinery, New York, NY, USA, 279–282. DOI:http://dx.doi.org/10.1145/1449715.1449761

[99] Chris Harrison and Scott E. Hudson. 2008b. Scratch input: creating large, inexpensive, unpowered and mobile finger input surfaces. In *Proceedings of the 21st annual ACM symposium on User interface software and technology (UIST '08)*. Association for Computing Machinery, New York, NY, USA, 205–208. DOI:http://dx.doi.org/10.1145/1449715.1449747

[100] Chris Harrison, Desney Tan, and Dan Morris. 2010. Skinput: appropriating the body as an input surface. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. Association for Computing Machinery, New York, NY, USA, 453–462. DOI:http://dx.doi.org/10.1145/1753326.1753394

[101] Anders Hast, Johan Nysjö, and Andrea Marchetti. 2013. Optimal RANSAC - Towards a Repeatable Algorithm for Finding the Optimal Set. *Journal of WSCG* 21 (2013), 10.

[102] Zhe He, Xiangyang Wang, Yuanchun Shi, Chi Hsia, Chen Liang, and Chun Yu. 2025. Palmpad: Enabling Real-Time Index-to-Palm Touch Interaction with a Single RGB Camera. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 551, 16 pages. DOI:http://dx.doi.org/10.1145/3706598.3714130

[103] Oura Health. 2024. Oura Ring. Smart Ring for Fitness, Stress, Sleep & Health. (2024). https://ouraring.com

[104] David Heaney. 2024. Quest 3S Has Better Low-Light Hand Tracking Than Quest 3. (Oct. 2024). https://www.uploadvr.com/quest3s-hand-tracking-better-than-quest-3/

[105] P Helliwell and W Taylor. 2004. Repetitive strain injury. *Postgraduate Medical Journal* 80, 946 (Aug. 2004), 438–443. DOI:http://dx.doi.org/10.1136/pgmj.2003.012591

[106] Dan Hendrycks and Kevin Gimpel. 2023. Gaussian Error Linear Units (GELUs). (June 2023). DOI:http://dx.doi.org/10.48550/arXiv.1606.08415 arXiv:1606.08415 [cs].

[107] Himax Imaging. 2020. Himax Launches HM0360 1/6" VGA Ultra Low Power CMOS Image Sensor for AIoT and Computer Vision Applications. Press release, GlobeNewswire. (Jan. 2020). https://www.globenewswire.com/news-release/2020/01/22/1973570/0/en/Himax-Launches-HM0360-1-6-VGA-Ultra-Low-Power-CMOS-Image-Sensor-for-AIoT-and-Computer-Visio
html Accessed: 2026-09-15.

[108] Jeroen D. Hol, Fred Dijkstra, Henk Luinge, and Thomas B. Schon. 2009. Tightly coupled UWB/IMU pose estimation. In *2009 IEEE International Conference on Ultra-Wideband*. 688–692. DOI:http://dx.doi.org/10.1109/ICUWB.2009.5288724

[109] 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 Camera. In *2021 IEEE International Conference on Image Processing (ICIP)*. 1304–1308. DOI:http://dx.doi.org/10.1109/ICIP42928.2021.9506043

[110] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. 2019. Searching for MobileNetV3. (Nov. 2019). DOI:http://dx.doi.org/10.48550/arXiv.1905.02244 arXiv:1905.02244 [cs.CV].

[111] Fang Hu, Peng He, Songlin Xu, Yin Li, and Cheng Zhang. 2020b. FingerTrak: Continuous 3D Hand Pose Tracking by Deep Learning Hand Silhouettes Captured by Miniature Thermal Cameras on Wrist. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 2 (June 2020), 71:1–71:24. DOI:http://dx.doi.org/10.1145/3397306

[112] Yuhan Hu, Sara Maria Bejarano, and Guy Hoffman. 2020a. ShadowSense: Detecting Human Touch in a Social Robot Using Shadow Image Classification. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4 (Dec. 2020), 132:1–132:24. DOI:http://dx.doi.org/10.1145/3432202

[113] De-An Huang, Minghuang Ma, Wei-Chiu Ma, and Kris M. Kitani. 2015. How do we use our hands? Discovering a diverse set of common grasps. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 666–675. DOI:http://dx.doi.org/10.1109/CVPR.2015.7298666 ISSN: 1063-6919.

[114] William Huang, Siyou Pei, Leyi Zou, Eric J Gonzalez, Ishan Chatterjee, and Yang Zhang. 2026. DeltaDorsal: Enhancing Hand Pose Estimation with Dorsal Features in Egocentric Views. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26)*. Association for Computing Machinery, New York, NY, USA, 1–16. DOI:http://dx.doi.org/10.1145/3772318.3790493

[115] Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J. Black, Otmar Hilliges, and Gerard Pons-Moll. 2018. Deep inertial poser: learning to reconstruct human pose from sparse inertial measurements in real time. *ACM Transactions on Graphics* 37, 6 (Dec. 2018), 1–15. DOI:http://dx.doi.org/10.1145/3272127.3275108

[116] HuatengVision. 2024. HT-SUA33GM-T1V-C : Huateng Vision. (2024). https://www.huatengvision.com/product/195/

[117] Humane. 2024. Humane Ai Pin | See the World, Not Your Screen. | Humane. (2024). https://humane.com/

[118] 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*. 98–111.

[119] Samuel A. Iacolina, Alessandro Soro, and Riccardo Scateni. 2011. Improving FTIR based multi-touch sensors with IR shadow tracking. In *Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems*. ACM, Pisa Italy, 241–246. DOI:http://dx.doi.org/10.1145/1996461.1996529

[120] Fumiko Ichikawa, Jan Chipchase, and R. Grignani. 2005. Where's the phone? A study of mobile phone location in public spaces. *Mobile Technology, Applications and Systems, 2005 2nd International Conference on* (Jan. 2005), 1–8.

[121] Apple Inc. 2022. Deploying Transformers on the Apple Neural Engine. (June 2022). https://machinelearning.apple.com/research/neural-engine-transformers

[122] Stephen S. Intille, Ling Bao, Emmanuel Munguia Tapia, and John Rondoni. 2004. Acquiring in situ training data for context-aware ubiquitous computing applications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04)*. Association for Computing Machinery, New York, NY, USA, 1–8. DOI:http://dx.doi.org/10.1145/985692.985693

[123] Yasha Iravantchi, Karan Ahuja, Mayank Goel, Chris Harrison, and Alanson Sample. 2021. PrivacyMic: Utilizing Inaudible Frequencies for Privacy Preserving Daily Activity Recognition. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, 1–13. DOI:http://dx.doi.org/10.1145/3411764.3445169

[124] Yasha Iravantchi, Mayank Goel, and Chris Harrison. 2019a. BeamBand: Hand Gesture Sensing with Ultrasonic Beamforming. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–10. DOI:http://dx.doi.org/10.1145/3290605.3300245

[125] Yasha Iravantchi, Yang Zhang, Evi Bernitsas, Mayank Goel, and Chris Harrison. 2019b. Interferi: Gesture Sensing using On-Body Acoustic Interferometry. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. DOI:http://dx.doi.org/10.1145/3290605.3300506

[126] Tatsuya Ishihara, Kris M. Kitani, Wei-Chiu Ma, Hironobu Takagi, and Chieko Asakawa. 2015. Recognizing hand-object interactions in wearable camera videos. In *2015 IEEE International Conference on Image Processing (ICIP)*. 1349–1353. DOI:http://dx.doi.org/10.1109/ICIP.2015.7351020

[127] Daisuke Iwai and Kosuke Sato. 2005. Heat sensation in image creation with thermal vision. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology (ACE '05)*. Association for Computing Machinery, New York, NY, USA, 213–216. DOI:http://dx.doi.org/10.1145/1178477.1178510

[128] Robert J.K. Jacob, Audrey Girouard, Leanne M. Hirshfield, Michael S. Horn, Orit Shaer, Erin Treacy Solovey, and Jamie Zigelbaum. 2008. Reality-Based Interaction: A Framework for Post-WIMP Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, 201–210. DOI:http://dx.doi.org/10.1145/1357054.1357089

[129] Silje Ekroll Jahren, Niels Aakvaag, Frode Strisland, Andreas Vogl, Alessandro Liberale, and Anders E. Liverud. 2021. Towards Human Motion Tracking Enhanced by Semi-Continuous Ultrasonic Time-of-Flight Measurements. *Sensors* 21, 7 (Jan. 2021), 2259. DOI:http://dx.doi.org/10.3390/s21072259

[130] Hao Jiang and Kristen Grauman. 2017. Seeing invisible poses: Estimating 3d body pose from egocentric video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 3501–3509.

[131] Jiaxi Jiang, Paul Streli, Huajian Qiu, Andreas Fender, Larissa Laich, Patrick Snape, and Christian Holz. 2022. AvatarPoser: Articulated Full-Body Pose Tracking from Sparse Motion Sensing. (July 2022). http://arxiv.org/abs/2207.13784 arXiv:2207.13784 [cs].

[132] Xianta Jiang, Lukas-Karim Merhi, and Carlo Menon. 2018. Force Exertion Affects Grasp Classification Using Force Myography. *IEEE Transactions on Human-Machine Systems* 48, 2 (April 2018), 219–226. DOI:http://dx.doi.org/10.1109/THMS.2017.2693245

[133] Haojian Jin, Zhijian Yang, Swarun Kumar, and Jason I. Hong. 2018. Towards Wearable Everyday Body-Frame Tracking using Passive RFIDs. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (Jan. 2018), 145:1–145:23. DOI:http://dx.doi.org/10.1145/3161199

[134] P. Jung, G. Lim, S. Kim, and K. Kong. 2015. A Wearable Gesture Recognition Device for Detecting Muscular Activities Based on Air-Pressure Sensors. *IEEE Transactions on Industrial Informatics* 11, 2 (April 2015), 485–494. DOI:http://dx.doi.org/10.1109/TII.2015.2405413 Conference Name: IEEE Transactions on Industrial Informatics.

[135] Hakan Karaoguz and Patric Jensfelt. 2019. Object Detection Approach for Robot Grasp Detection. In *2019 International Conference on Robotics and Automation (ICRA)*. 4953–4959. DOI:http://dx.doi.org/10.1109/ICRA.2019.8793751 ISSN: 2577-087X.

[136] Kanchan Keisham, Thenukan Pathmanathan, and Thangarajah Akilan. 2026. ConTrans: Learning Text-enhanced Local-global Temporal Representations for Zero-shot Temporal Action Localization. (May 2026). DOI:http://dx.doi.org/10.48550/arXiv.2605.30689 arXiv:2605.30689 [cs.CV].

[137] Frederic Kerber, Michael Puhl, and Antonio Krüger. 2017. User-Independent Real-Time Hand Gesture Recognition Based on Surface Electromyography. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '17)*. Association for Computing Machinery, New York, NY, USA, Article 36, 7 pages. DOI:http://dx.doi.org/10.1145/3098279.3098553

[138] Wolf Kienzle and Ken Hinckley. 2014. LightRing: always-available 2D input on any surface. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. ACM, Honolulu Hawaii USA, 157–160. DOI:http://dx.doi.org/10.1145/2642918.2647376

[139] Wolf Kienzle, Eric Whitmire, Chris Rittaler, and Hrvoje Benko. 2021. ElectroRing: Subtle Pinch and Touch Detection with a Ring. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–12. DOI:http://dx.doi.org/10.1145/3411764.3445094

[140] David Kim, Otmar Hilliges, Shahram Izadi, Alex D. Butler, Jiawen Chen, Iason Oikonomidis, and Patrick Olivier. 2012. Digits: freehand 3D interactions anywhere using a wrist-worn gloveless sensor. In *Proceedings of the 25th annual ACM symposium on User interface software and technology (UIST '12)*. Association for Computing Machinery, New York, NY, USA, 167–176. DOI:http://dx.doi.org/10.1145/2380116.2380139

[141] Daehwa Kim, Keunwoo Park, and Geehyuk Lee. 2021. AtaTouch: Robust Finger Pinch Detection for a VR Controller Using RF Return Loss. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 11, 9 pages. DOI:http://dx.doi.org/10.1145/3411764.3445442

[142] Daehwa Kim, Eric Whitmire, Roger Boldu, Wolf Kienzle, and Hrvoje Benko. 2024. SoundScroll: Robust Finger Slide Detection Using Friction Sound and Wrist-Worn Microphones. In *Proceedings of the 2024 ACM International Symposium on Wearable Computers (ISWC '24)*. Association for Computing Machinery, New York, NY, USA, 63–70. DOI:http://dx.doi.org/10.1145/3675095.3676614

[143] Maruchi Kim, Antonio Glenn, Bandhav Veluri, Yunseo Lee, Eyoel Gebre, Aditya Bagaria, Shwetak Patel, and Shyamnath Gollakota. 2024. IRIS: Wireless ring for vision-based smart home interaction. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. ACM, Pittsburgh PA USA, 1–16. DOI:http://dx.doi.org/10.1145/3654777.3676327

[144] Jung In Koh, Josh Cherian, Paul Taele, and Tracy Hammond. 2019. Developing a Hand Gesture Recognition System for Mapping Symbolic Hand Gestures to Analogous Emojis in Computer-Mediated Communication. *ACM Trans. Interact. Intell. Syst.* 9, 1, Article 6 (March 2019), 35 pages. DOI:http://dx.doi.org/10.1145/3297277

[145] Myron W. Krueger, Thomas Gionfriddo, and Katrin Hinrichsen. 1985. VIDEOPLACE—an artificial reality. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '85)*. Association for Computing Machinery, New York, NY, USA, 35–40. DOI:http://dx.doi.org/10.1145/317456.317463

[146] Anna Kukleva, Hilde Kuehne, Fadime Sener, and Juergen Gall. 2019. Unsupervised learning of action classes with continuous temporal embedding. (April 2019). DOI:http://dx.doi.org/10.48550/arXiv.1904.04189 arXiv:1904.04189 [cs.CV].

[147] Daniel Kurz. 2014. Thermal touch: Thermography-enabled everywhere touch interfaces for mobile augmented reality applications. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 9–16. DOI:http://dx.doi.org/10.1109/ISMAR.2014.6948403

[148] Thalmic Labs. 2016. Unveiling the Final Design of the Myo™ Armband. (Sept. 2016). https://medium.com/thalmic/unveiling-the-final-design-of-the-myo-armband-10576c0ae95b

[149] Nicholas D. Lane, Petko Georgiev, and Lorena Qendro. 2015. DeepEar: robust smartphone audio sensing in unconstrained acoustic environments using deep learning. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 283–294. DOI:http://dx.doi.org/10.1145/2750858.2804262

[150] Gierad Laput, Karan Ahuja, Mayank Goel, and Chris Harrison. 2018. Ubicoustics: Plug-and-Play Acoustic Activity Recognition. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. Association for Computing Machinery, New York, NY, USA, 213–224. DOI:http://dx.doi.org/10.1145/3242587.3242609

[151] Gierad Laput and Chris Harrison. 2019a. Sensing Fine-Grained Hand Activity with Smartwatches. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. DOI:http://dx.doi.org/10.1145/3290605.3300568

[152] Gierad Laput and Chris Harrison. 2019b. 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 (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. DOI:http://dx.doi.org/10.1145/3290605.3300559

[153] Gierad Laput, Robert Xiao, and Chris Harrison. 2016. ViBand: High-Fidelity Bio-Acoustic Sensing Using Commodity Smartwatch Accelerometers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. Association for Computing Machinery, New York, NY, USA, 321–333. DOI:http://dx.doi.org/10.1145/2984511.2984582

[154] Gierad Laput, Chouchang Yang, Robert Xiao, Alanson Sample, and Chris Harrison. 2015. EM-Sense: Touch Recognition of Uninstrumented, Electrical and Electromechanical Objects. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. Association for Computing Machinery, New York, NY, USA, 157–166. DOI:http://dx.doi.org/10.1145/2807442.2807481

[155] Eric Larson, Gabe Cohn, Sidhant Gupta, Xiaofeng Ren, Beverly Harrison, Dieter Fox, and Shwetak Patel. 2011. HeatWave: thermal imaging for surface user interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. Association for Computing Machinery, New York, NY, USA, 2565–2574. DOI:http://dx.doi.org/10.1145/1978942.1979317

[156] Chi-Jung Lee, Jiaxin Li, Tianhong Catherine Yu, Ruidong Zhang, Vipin Gunda, François Guimbretière, and Cheng Zhang. 2025. Grab-n-Go: On-the-Go Microgesture Recognition with Objects in Hand. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 9, 3, Article 99 (Sept. 2025), 27 pages. DOI:http://dx.doi.org/10.1145/3749469

[157] Chi-Jung Lee, Ruidong Zhang, Devansh Agarwal, Tianhong Catherine Yu, Vipin Gunda, Oliver Lopez, James Kim, Sicheng Yin, Boao Dong, Ke Li, Mose Sakashita, Francois Guimbretiere, and Cheng Zhang. 2024. EchoWrist: Continuous Hand Pose Tracking and Hand-Object Interaction Recognition Using Low-Power Active Acoustic Sensing On a Wristband. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 403, 21 pages. DOI:http://dx.doi.org/10.1145/3613904.3642910

[158] Boning Li and Akane Sano. 2020. Extraction and Interpretation of Deep Autoencoder-based Temporal Features from Wearables for Forecasting Personalized Mood, Health, and Stress. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 2 (June 2020), 49:1–49:26. DOI:http://dx.doi.org/10.1145/3397318

[159] Chen Liang, Xutong Wang, Zisu Li, Chi Hsia, Mingming Fan, Chun Yu, and Yuanchun Shi. 2023. ShadowTouch: Enabling Free-Form Touch-Based Hand-to-Surface Interaction with Wrist-Mounted Illuminant by Shadow Projection. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23)*. Association for Computing Machinery, New York, NY, USA, 1–14. DOI:http://dx.doi.org/10.1145/3586183.3606785

[160] Chen Liang, Chun Yu, Yue Qin, Yuntao Wang, and Yuanchun Shi. 2021. DualRing: Enabling Subtle and Expressive Hand Interaction with Dual IMU Rings. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 3 (Sept. 2021), 1–27. DOI:http://dx.doi.org/10.1145/3478114

[161] Dawei Liang, Guihong Li, Rebecca Adaimi, Radu Marculescu, and Edison Thomaz. 2022. AudioIMU: Enhancing Inertial Sensing-Based Activity Recognition with Acoustic Models. In *Proceedings of the 2022 ACM International Symposium on Wearable Computers (ISWC '22)*. Association for Computing Machinery, New York, NY, USA, 44–48. DOI:http://dx.doi.org/10.1145/3544794.3558471

[162] Life360. 2022. Tile Trackers: Bluetooth Wallet & Key Trackers | Life360. (2022). https://www.life360.com/tile-trackers

[163] 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. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 3 (Sept. 2022), 1–21. DOI:http://dx.doi.org/10.1145/3552312

[164] Wonseon Lim, Hyejeong Im, and Dae-Won Kim. 2026. Continual Multimodal Egocentric Activity Recognition via Modality-Aware Novel Detection. (2026). DOI:http://dx.doi.org/10.48550/ARXIV.2603.16970 Version Number: 1.

[165] Jhe-Wei Lin, Chiuan Wang, Yi Yao Huang, Kuan-Ting Chou, Hsuan-Yu Chen, Wei-Luan Tseng, and Mike Y. Chen. 2015. BackHand: Sensing Hand Gestures via Back of the Hand. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. Association for Computing Machinery, New York, NY, USA, 557–564. DOI:http://dx.doi.org/10.1145/2807442.2807462

[166] Shu-Yang Lin, Chao-Huai Su, Kai-Yin Cheng, Rong-Hao Liang, Tzu-Hao Kuo, and Bing-Yu Chen. 2011. Pub - point upon body: exploring eyes-free interaction and methods on an arm. In *Proceedings of the 24th annual ACM symposium on User interface software and technology (UIST '11)*. Association for Computing Machinery, New York, NY, USA, 481–488. DOI:http://dx.doi.org/10.1145/2047196.2047259

[167] Yizhou Lin, Gang Hua, and Philippos Mordohai. 2015. Egocentric Object Recognition Leveraging the 3D Shape of the Grasping Hand. In *Computer Vision - ECCV 2014 Workshops*, Lourdes Agapito, Michael M. Bronstein, and Carsten Rother (Eds.). Springer International Publishing, Cham, 746–762. DOI:http://dx.doi.org/10.1007/978-3-319-16199-0_52

[168] Tommaso Lisini Baldi, Francesco Farina, Andrea Garulli, Antonio Giannitrapani, and Domenico Prattichizzo. 2020. Upper Body Pose Estimation Using Wearable Inertial Sensors and Multiplicative Kalman Filter. *IEEE Sensors Journal* 20, 1 (Jan. 2020), 492–500. DOI:http://dx.doi.org/10.1109/JSEN.2019.2940612 Conference Name: IEEE Sensors Journal.

[169] Huajun Liu, Xiaolin Wei, Jinxiang Chai, Inwoo Ha, and Taehyun Rhee. 2011. Realtime human motion control with a small number of inertial sensors. In *Symposium on Interactive 3D Graphics and Games (I3D '11)*. Association for Computing Machinery, New York, NY, USA, 133–140. DOI:http://dx.doi.org/10.1145/1944745.1944768

[170] Mengxi Liu, Bo Zhou, Vitor Fortes Rey, Sizhen Bian, and Paul Lukowicz. 2024. iEat: automatic wearable dietary monitoring with bio-impedance sensing. *Scientific Reports* 14 (Aug. 2024), 17873. DOI:http://dx.doi.org/10.1038/s41598-024-67765-5

[171] Shi Qiang Liu, Jun Chang Zhang, and Rong Zhu. 2020. A Wearable Human Motion Tracking Device Using Micro Flow Sensor Incorporating a Micro Accelerometer. *IEEE Transactions on Biomedical Engineering* 67, 4 (April 2020), 940–948. DOI:http://dx.doi.org/10.1109/TBME.2019.2924689 Conference Name: IEEE Transactions on Biomedical Engineering.

[172] Yang Liu, Zhenjiang Li, Zhidan Liu, and Kaishun Wu. 2019. Real-time Arm Skeleton Tracking and Gesture Inference Tolerant to Missing Wearable Sensors. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '19)*. Association for Computing Machinery, New York, NY, USA, 287–299. DOI:http://dx.doi.org/10.1145/3307334.3326109

[173] Yilin Liu, Shijia Zhang, and Mahanth Gowda. 2021. NeuroPose: 3D Hand Pose Tracking Using EMG Wearables. In *Proceedings of the Web Conference 2021 (WWW '21)*. Association for Computing Machinery, New York, NY, USA, 1471–1482. DOI:http://dx.doi.org/10.1145/3442381.3449890

[174] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. 2019. MediaPipe: A Framework for Building Perception Pipelines. (2019). DOI:http://dx.doi.org/10.48550/ARXIV.1906.08172

[175] Minghuang Ma, Haoqi Fan, and Kris M. Kitani. 2016. Going Deeper into First-Person Activity Recognition. (May 2016). DOI:http://dx.doi.org/10.48550/arXiv.1605.03688 arXiv:1605.03688 [cs.CV].

[176] Takuya Maekawa, Yasue Kishino, Yasushi Sakurai, and Takayuki Suyama. 2011. Recognizing the Use of Portable Electrical Devices with Hand-Worn Magnetic Sensors. In *Pervasive Computing (Lecture Notes in Computer Science)*, Kent Lyons, Jeffrey Hightower, and Elaine M. Huang (Eds.). Springer, Berlin, Heidelberg, 276–293. DOI:http://dx.doi.org/10.1007/978-3-642-21726-5_18

[177] Takuya Maekawa, Yasue Kishino, Yutaka Yanagisawa, and Yasushi Sakurai. 2012a. Recognizing Handheld Electrical Device Usage with Hand-Worn Coil of Wire. In *Pervasive Computing (Lecture Notes in Computer Science)*, Judy Kay, Paul Lukowicz, Hideyuki Tokuda, Patrick Olivier, and Antonio Krüger (Eds.). Springer, Berlin, Heidelberg, 234–252. DOI:http://dx.doi.org/10.1007/978-3-642-31205-2_15

[178] Takuya Maekawa, Yasue Kishino, Yutaka Yanagisawa, and Yasushi Sakurai. 2012b. WristSense: Wrist-worn sensor device with camera for daily activity recognition. In *2012 IEEE International Conference on Pervasive Computing and Communications Workshops*. 510–512. DOI:http://dx.doi.org/10.1109/PerComW.2012.6197551

[179] Takuya Maekawa, Yutaka Yanagisawa, Yasue Kishino, Katsuhiko Ishiguro, Koji Kamei, Yasushi Sakurai, and Takeshi Okadome. 2010. Object-Based Activity Recognition with Heterogeneous Sensors on Wrist. In *Pervasive Computing (Lecture Notes in Computer Science)*, Patrik Floréen, Antonio Krüger, and Mirjana Spasojevic (Eds.). Springer, Berlin, Heidelberg, 246–264. DOI:http://dx.doi.org/10.1007/978-3-642-12654-3_15

[180] Michele Magno, Lukas Cavigelli, Renzo Andri, and Luca Benini. 2016. Ultra-Low Power Context Recognition Fusing Sensor Data from an Energy-Neutral Smart Watch. In *Internet of*

Things. IoT Infrastructures, Benny Mandler, Johann Marquez-Barja, Miguel Elias Mitre Campista, Dagmar Cagáňová, Hakima Chaouchi, Sherali Zeadally, Mohamad Badra, Stefano Giordano, Maria Fazio, Andrey Somov, and Radu-Laurentiu Vieriu (Eds.). Springer International Publishing, Cham, 331–343. DOI:http://dx.doi.org/10.1007/978-3-319-47075-7_38

[181] Saif Mahmud, Ke Li, Guilin Hu, Hao Chen, Richard Jin, Ruidong Zhang, François Guimbretière, and Cheng Zhang. 2023. PoseSonic: 3D Upper Body Pose Estimation Through Egocentric Acoustic Sensing on Smartglasses. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 3 (Sept. 2023), 111:1–111:28. DOI:http://dx.doi.org/10.1145/3610895

[182] Saif Mahmud, Vineet Parikh, Qikang Liang, Ke Li, Ruidong Zhang, Ashwin Ajit, Vipin Gunda, Devansh Agarwal, François Guimbretière, and Cheng Zhang. 2024. ActSonic: Recognizing Everyday Activities from Inaudible Acoustic Wave Around the Body. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 4 (Nov. 2024), 1–32. DOI:http://dx.doi.org/10.1145/3699752 arXiv:2404.13924 [cs.HC].

[183] J.L. Marins, Xiaoping Yun, E.R. Bachmann, R.B. McGhee, and M.J. Zyda. 2001. An extended Kalman filter for quaternion-based orientation estimation using MARG sensors. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the Next Millennium (Cat. No.01CH37180)*, Vol. 4. IEEE, Maui, HI, USA, 2003–2011. DOI:http://dx.doi.org/10.1109/IROS.2001.976367

[184] Joe Marshall, Tony Pridmore, Mike Pound, Steve Benford, and Boriana Koleva. 2009. Pressing the Flesh: Sensing Multiple Touch and Finger Pressure on Arbitrary Surfaces. In *Proceedings of the 6th International Conference on Pervasive Computing (Pervasive '08)*. Springer-Verlag, Berlin, Heidelberg, 38–55. DOI:http://dx.doi.org/10.1007/978-3-540-79576-6_3

[185] Damien Masson, Alix Goguey, Sylvain Malacria, and Géry Casiez. 2017. WhichFingers: Identifying Fingers on Touch Surfaces and Keyboards using Vibration Sensors. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. Association for Computing Machinery, New York, NY, USA, 41–48. DOI:http://dx.doi.org/10.1145/3126594.3126619

[186] Takashi Matsubara, Naoki Mori, Takehiro Niikura, and Shun'ichi Tano. 2017. Touch detection method for non-display surface using multiple shadows of finger. In *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)*. 1–5. DOI:http://dx.doi.org/10.1109/GCCE.2017.8229364

[187] Christophe Maufroy and Daniel Bargmann. 2018. CNN-Based Detection and Classification of Grasps Relevant for Worker Support Scenarios Using sEMG Signals of Forearm Muscles. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, Miyazaki, Japan, 141–146. DOI:http://dx.doi.org/10.1109/SMC.2018.00035

[188] Jess McIntosh, Asier Marzo, and Mike Fraser. 2017a. SensIR: Detecting Hand Gestures with a Wearable Bracelet using Infrared Transmission and Reflection. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. Association for Computing Machinery, New York, NY, USA, 593–597. DOI:http://dx.doi.org/10.1145/3126594.3126604

[189] Jess McIntosh, Asier Marzo, Mike Fraser, and Carol Phillips. 2017b. EchoFlex: Hand Gesture Recognition Using Ultrasound Imaging. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, New York, NY, USA, 1923–1934. DOI:http://dx.doi.org/10.1145/3025453.3025807

[190] Manuel Meier, Paul Streli, Andreas Fender, and Christian Holz. 2021. TapID: Rapid Touch Interaction in Virtual Reality using Wearable Sensing. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*. 519–528. DOI:http://dx.doi.org/10.1109/VR50410.2021.00076 ISSN: 2642-5254.

[191] Zemene W. Mekonnen, Eric Slottke, Heinrich Luecken, Christoph Steiner, and Armin Wittneben. 2010. Constrained maximum likelihood positioning for UWB based human motion tracking. In *2010 International Conference on Indoor Positioning and Indoor Navigation*. 1–10. DOI:http://dx.doi.org/10.1109/IPIN.2010.5647912

[192] J. A. Mercer, N. Bezodis, D. DeLion, T. Zachry, and M. D. Rubley. 2006. EMG sensor location: Does it influence the ability to detect differences in muscle contraction conditions? *Journal of Electromyography and Kinesiology* 16, 2 (April 2006), 198–204. DOI:http://dx.doi.org/10.1016/j.jelekin.2005.07.002

[193] Meta. 2023. Meta Quest 3: Mixed Reality VR Headset - Shop Now. (2023). https://www.meta.com/quest/quest-3/

[194] Meta. 2026. Battery life on AI glasses | AI Glasses Help. (2026). https://www.meta.com/help/ai-glasses/303057485648146/

[195] Jochen Meyer, Jutta Fortmann, Merlin Wasmann, and Wilko Heuten. 2015. Making Lifelogging Usable: Design Guidelines for Activity Trackers. In *MultiMedia Modeling (Lecture Notes in Computer Science)*, Xiangjian He, Suhuai Luo, Dacheng Tao, Changsheng Xu, Jie Yang, and Muhammad Abul Hasan (Eds.). Springer International Publishing, Cham, 323–334. DOI:http://dx.doi.org/10.1007/978-3-319-14442-9_39

[196] Johannes Meyer, Adrian Frank, Thomas Schlebusch, and Enkeljeda Kasneci. 2022. A CNN-based Human Activity Recognition System Combining a Laser Feedback Interferometry Eye Movement Sensor and an IMU for Context-aware Smart Glasses. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 4 (Dec. 2022), 172:1–172:24. DOI:http://dx.doi.org/10.1145/3494998

[197] Markus Miezal, Gabriele Bleser, Norbert Schmitz, and Didier Stricker. 2013. A generic approach to inertial tracking of arbitrary kinematic chains. In *Proceedings of the 8th International Conference on Body Area Networks (BodyNets '13)*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Brussels, BEL, 189–192. DOI:http://dx.doi.org/10.4108/icst.bodynets.2013.253608

[198] Christen Millerdurai, Shaoxiang Wang, Yaxu Xie, Vladislav Golyanik, Didier Stricker, and Alain Pagani. 2026. EgoForce: Forearm-Guided Camera-Space 3D Hand Pose from a Monocular Egocentric Camera. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Papers (SIGGRAPH Conference Papers '26)*. Association for Computing Machinery, New York, NY, USA, 1–12. DOI:http://dx.doi.org/10.1145/3799902.3811047

[199] Kyle Min and Jason J. Corso. 2021. Integrating Human Gaze into Attention for Egocentric Activity Recognition. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Waikoloa, HI, USA, 1068–1077. DOI:http://dx.doi.org/10.1109/WACV48630.2021.00111

[200] Vimal Mollyn, Karan Ahuja, Dhruv Verma, Chris Harrison, and Mayank Goel. 2022. SAMoSA: Sensing Activities with Motion and Subsampled Audio. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 3 (Sept. 2022), 132:1–132:19. DOI:http://dx.doi.org/10.1145/3550284

[201] Vimal Mollyn, Riku Arakawa, Mayank Goel, Chris Harrison, and Karan Ahuja. 2023. IMUPoser: Full-Body Pose Estimation using IMUs in Phones, Watches, and Earbuds. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, 1–12. DOI:http://dx.doi.org/10.1145/3544548.3581392

[202] Vimal Mollyn, Nathan DeVrio, and Chris Harrison. 2025. EclipseTouch: Touch Segmentation on Ad Hoc Surfaces using Worn Infrared Shadow Casting. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, New York, NY, USA. DOI:http://dx.doi.org/10.1145/3746059.3747743

[203] Vimal Mollyn and Chris Harrison. 2024. EgoTouch: On-Body Touch Input Using AR/VR Headset Cameras. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24)*. Association for Computing Machinery, New York, NY, USA, 1–11. DOI:http://dx.doi.org/10.1145/3654777.3676455

[204] Alessandra Moschetti, Laura Fiorini, Dario Esposito, Paolo Dario, and Filippo Cavallo. 2017. Daily activity recognition with inertial ring and bracelet: An unsupervised approach. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 3250–3255. DOI:http://dx.doi.org/10.1109/ICRA.2017.7989370

[205] Adiyan Mujibiya, Xiang Cao, Desney S. Tan, Dan Morris, Shwetak N. Patel, and Jun Rekimoto. 2013. The sound of touch: on-body touch and gesture sensing based on transdermal ultrasound propagation. In *Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces (ITS '13)*. Association for Computing Machinery, New York, NY, USA, 189–198. DOI:http://dx.doi.org/10.1145/2512349.2512821

[206] Andrea B. Neiman. 2017. CDC Grand Rounds: Improving Medication Adherence for Chronic Disease Management — Innovations and Opportunities. *MMWR. Morbidity and Mortality Weekly Report* 66 (2017). DOI:http://dx.doi.org/10.15585/mmwr.mm6645a2

[207] Kim Doang Nguyen, I-Ming Chen, Zhiqiang Luo, Song Huat Yeo, and Henry Been-Lirn Duh. 2011. A Wearable Sensing System for Tracking and Monitoring of Functional Arm Movement. *IEEE/ASME Transactions on Mechatronics* 16, 2 (April 2011), 213–220. DOI:http://dx.doi.org/10.1109/TMECH.2009.2039222 Conference Name: IEEE/ASME Transactions on Mechatronics.

[208] Takehiro Niikura, Takashi Matsubara, and Naoki Mori. 2016. Touch Detection System for Various Surfaces Using Shadow of Finger. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces*. ACM, Niagara Falls Ontario Canada, 337–342. DOI:http://dx.doi.org/10.1145/2992154.2996777

[209] Takehiro Niikura, Yoshihiro Watanabe, and Masatoshi Ishikawa. 2014. Anywhere surface touch: utilizing any surface as an input area. In *Proceedings of the 5th Augmented Human International Conference (AH '14)*. Association for Computing Machinery, New York, NY, USA, 1–8. DOI:http://dx.doi.org/10.1145/2582051.2582090

[210] Ju Young Oh, Jun Lee, Joong Ho Lee, and Ji Hyung Park. 2017. AnywhereTouch: Finger Tracking Method on Arbitrary Surface Using Nailed-Mounted IMU for Mobile HMD. In *HCI International 2017 – Posters' Extended Abstracts*, Constantine Stephanidis (Ed.). Springer International Publishing, Cham, 185–191. DOI:http://dx.doi.org/10.1007/978-3-319-58750-9_26

[211] Ju Young Oh, Ji-Hyung Park, and Jung-Min Park. 2020. FingerTouch: Touch Interaction Using a Fingernail-Mounted Sensor on a Head-Mounted Display for Augmented Reality. *IEEE Access* 8 (2020), 101192–101208. DOI:http://dx.doi.org/10.1109/ACCESS.2020.2997972 Conference Name: IEEE Access.

[212] Seungjae Oh, Gyeore Yun, Chaeyong Park, Jinsoo Kim, and Seungmoon Choi. 2019. VibEye: Vibration-Mediated Object Recognition for Tangible Interactive Applications. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. DOI:http://dx.doi.org/10.1145/3290605.3300906

[213] Takehiko Ohkawa, Kun He, Fadime Sener, Tomas Hodan, Luan Tran, and Cem Keskin. 2023. AssemblyHands: Towards Egocentric Activity Understanding via 3D Hand Pose Estimation. (April 2023). DOI:http://dx.doi.org/10.48550/arXiv.2304.12301 arXiv:2304.12301 [cs.CV].

[214] Katsunori Ohnishi, Atsushi Kanehira, Asako Kanezaki, and Tatsuya Harada. 2016. Recognizing Activities of Daily Living with a Wrist-Mounted Camera. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Las Vegas, NV, USA, 3103–3111. DOI: http://dx.doi.org/10.1109/CVPR.2016.338

[215] Makoto Ono, Buntarou Shizuki, and Jiro Tanaka. 2013. Touch & activate: adding interactivity to existing objects using active acoustic sensing. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. ACM, St. Andrews Scotland, United Kingdom, 31–40. DOI:http://dx.doi.org/10.1145/2501988.2501989

[216] Francisco Javier Ordóñez and Daniel Roggen. 2016. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors* 16, 1 (Jan. 2016), 115. DOI:http://dx.doi.org/10.3390/s16010115

[217] Santiago Ortega-Avila, Bogdana Rakova, Sajid Sadi, and Pranav Mistry. 2015. Non-invasive optical detection of hand gestures. In *Proceedings of the 6th Augmented Human International Conference (AH '15)*. Association for Computing Machinery, New York, NY, USA, 179–180. DOI:http://dx.doi.org/10.1145/2735711.2735801

[218] Pak-Kiu Chung, Bing Fang, and F. Quek. 2008. MirrorTrack - a vision based multi-touch system for glossy display surfaces. In *5th International Conference on Visual Information Engineering (VIE 2008)*. IEE, Xi'an, China, 571–576. DOI:http://dx.doi.org/10.1049/cp:20080379

[219] Anupam Pani and Yanchao Yang. 2026. Gaze-Regularized VLMs for Ego-Centric Behavior Understanding. (March 2026). https://arxiv.org/abs/2603.23190v1

[220] Vineet Parikh, Saif Mahmud, Devansh Agarwal, Ke Li, François Guimbretière, and Cheng Zhang. 2024. EchoGuide: Active Acoustic Guidance for LLM-Based Eating Event Analysis from Egocentric Videos. (July 2024). DOI:http://dx.doi.org/10.48550/arXiv.2406.10750 arXiv:2406.10750 [cs.HC].

[221] Siyou Pei, Pradyumna Chari, Xue Wang, Xiaoying Yang, Achuta Kadambi, and Yang Zhang. 2022. ForceSight: Non-Contact Force Sensing with Laser Speckle Imaging. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology (UIST '22)*. Association for Computing Machinery, New York, NY, USA, 1–11. DOI:http://dx.doi.org/10.1145/3526113.3545622

[222] DT Pham, M Al-Kutubi, Z Ji, M Yang, Z Wang, and S Catheline. 2005. Tangible acoustic interface approaches. In *Proceedings of IPROMS 2005 Virtual Conference*. Citeseer, 497–502.

[223] Angkoon Phinyomark and Erik Scheme. 2018. EMG Pattern Recognition in the Era of Big Data and Deep Learning. *Big Data and Cognitive Computing* 2, 3 (Sept. 2018), 21. DOI:http://dx.doi.org/10.3390/bdcc2030021 Number: 3 Publisher: Multidisciplinary Digital Publishing Institute.

[224] Tobias Pistohl, Thomas Sebastian Benedikt Schmidt, Tonio Ball, Andreas Schulze-Bonhage, Ad Aertsen, and Carsten Mehring. 2013. Grasp Detection from Human ECoG during Natural Reach-to-Grasp Movements. *PLOS ONE* 8, 1 (Jan. 2013), e54658. DOI:http://dx.doi.org/10.1371/journal.pone.0054658 Publisher: Public Library of Science.

[225] Polhemus. 2020. Polhemus motion tracking. (2020). https://polhemus.com

[226] Erez Posner, Nick Starzicki, and Eyal Katz. 2012. A single camera based floating virtual keyboard with improved touch detection. In *2012 IEEE 27th Convention of Electrical and Electronics Engineers in Israel*. 1–5. DOI:http://dx.doi.org/10.1109/EEEI.2012.6377072

[227] Will Price, Carl Vondrick, and Dima Damen. 2022. UnweaveNet: Unweaving Activity Stories. (April 2022). DOI:http://dx.doi.org/10.48550/arXiv.2112.10194 arXiv:2112.10194 [cs.CV].

[228] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. 2000. The Cricket Location-Support System. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom '00)*. Association for Computing Machinery, New York, NY, USA, 32–43. DOI:http://dx.doi.org/10.1145/345910.345917

[229] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2013. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th annual international conference on Mobile computing & networking (MobiCom '13)*. Association for Computing Machinery, New York, NY, USA, 27–38. DOI:http://dx.doi.org/10.1145/2500423.2500436

[230] Thai Duy Quy, Chih-Yang Lin, and Timothy K. Shih. 2025. Enhanced Human Activity Recognition Using Wi-Fi Sensing: Leveraging Phase and Amplitude with Attention Mechanisms. *Sensors* 25, 4 (Jan. 2025), 1038. DOI:http://dx.doi.org/10.3390/s25041038

[231] Frederick H. Raab, Ernest B. Blood, Terry O. Steiner, and Herbert R. Jones. 1979. Magnetic Position and Orientation Tracking System. *IEEE Trans. Aerospace Electron. Systems* AES-15, 5 (Sept. 1979), 709–718. DOI:http://dx.doi.org/10.1109/TAES.1979.308860 Conference Name: IEEE Transactions on Aerospace and Electronic Systems.

[232] Valentin Radu, Catherine Tong, Sourav Bhattacharya, Nicholas D. Lane, Cecilia Mascolo, Mahesh K. Marina, and Fahim Kawsar. 2018. Multimodal Deep Learning for Activity and Context Recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (Jan. 2018), 157:1–157:27. DOI:http://dx.doi.org/10.1145/3161174

[233] Vijay Rajanna, Raniero Lara-Garduno, Dev Jyoti Behera, Karthic Madanagopal, Daniel Goldberg, and Tracy Hammond. 2014. Step up life: a context aware health assistant. In *Proceedings of the Third ACM SIGSPATIAL International Workshop on the Use of GIS in Public Health (HealthGIS '14)*. Association for Computing Machinery, New York, NY, USA, 21–30. DOI:http://dx.doi.org/10.1145/2676629.2676636

[234] Mary Redmayne. 2017. Where's Your Phone? A Survey of Where Women Aged 15-40 Carry Their Smartphone and Related Risk Perception: A Survey and Pilot Study. *PLoS ONE* 12, 1 (Jan. 2017), e0167996. DOI:http://dx.doi.org/10.1371/journal.pone.0167996

[235] Blaine Reeder. 2016. Health at hand: A systematic review of smart watch uses for health and wellness. *Journal of Biomedical Informatics* (2016), 8.

[236] J. Rekimoto. 2001. GestureWrist and GesturePad: unobtrusive wearable interaction devices. In *Proceedings Fifth International Symposium on Wearable Computers*. 21–27. DOI:http://dx.doi.org/10.1109/ISWC.2001.962092 ISSN: 1530-0811.

[237] Yili Ren, Zi Wang, Sheng Tan, Yingying Chen, and Jie Yang. 2022a. Winect: 3D Human Pose Tracking for Free-form Activity Using Commodity WiFi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 4 (Dec. 2022), 176:1–176:29. DOI:http://dx.doi.org/10.1145/3494973

[238] Yili Ren, Zi Wang, Yichao Wang, Sheng Tan, Yingying Chen, and Jie Yang. 2022b. GoPose: 3D Human Pose Estimation Using WiFi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 2 (July 2022), 69:1–69:25. DOI:http://dx.doi.org/10.1145/3534605

[239] Mark Richardson, Matt Durasoff, and Robert Wang. 2020. Decoding Surface Touch Typing from Hand-Tracking. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. ACM, Virtual Event USA, 686–696. DOI:http://dx.doi.org/10.1145/3379337.3415816

[240] Daniel Roetenberg, Henk Luinge, and Per Slycke. 2013. Xsens MVN: Full 6DOF Human Motion Tracking Using Miniature Inertial Sensors. (2013), 9.

[241] Daniel Roetenberg, Per J. Slycke, and Peter H. Veltink. 2007. Ambulatory Position and Orientation Tracking Fusing Magnetic and Inertial Sensing. *IEEE Transactions on Biomedical Engineering* 54, 5 (May 2007), 883–890. DOI:http://dx.doi.org/10.1109/TBME.2006.889184 Conference Name: IEEE Transactions on Biomedical Engineering.

[242] J. Rudolph, Valkyrie Savage, Ricardo Jota, B. Araújo, David Holman, and Daniel J. Wigdor. 2021. Hand as Sensor: Virtualizing Interactions with Everyday Object Affordances using Wrist-worn Hand Pose Sensing.

[243] Julius Cosmo Romeo Rudolph, David Holman, Bruno De Araujo, Ricardo Jota, Daniel Wigdor, and Valkyrie Savage. 2022. Sensing Hand Interactions with Everyday Objects by Profiling Wrist Topography. In *Sixteenth International Conference on Tangible, Embedded, and Embodied Interaction*. ACM, Daejeon Republic of Korea, 1–14. DOI:http://dx.doi.org/10.1145/3490149.3501320

[244] K. Martin Sagayam and D. Jude Hemanth. 2017. Hand posture and gesture recognition techniques for virtual reality applications: a survey. *Virtual Reality* 21, 2 (June 2017), 91–107. DOI:http://dx.doi.org/10.1007/s10055-016-0301-0

[245] Farshid Salemi Parizi, Wolf Kienzle, Eric Whitmire, Aakar Gupta, and Hrvoje Benko. 2021. RotoWrist: Continuous Infrared Wrist Angle Tracking using a Wristband. In *Proceedings of the 27th ACM Symposium on Virtual Reality Software and Technology*. ACM, Osaka Japan, 1–11. DOI:http://dx.doi.org/10.1145/3489849.3489886

[246] Anik Sarker, Ziyi Kou, Ergys Ristani, Li Guan, and Taylor Niehues. 2026. Real-Time Hand Pose Tracking using 6-Axis IMUs. In *Proceedings of the 21st ACM/IEEE International Conference on Human-Robot Interaction (HRI '26)*. Association for Computing Machinery, New York, NY, USA, 1182–1191. DOI:http://dx.doi.org/10.1145/3757279.3785628

[247] A. Schmidt, H.-W. Gellersen, and C. Merz. 2000. Enabling implicit human computer interaction: a wearable RFID-tag reader. In *Digest of Papers. Fourth International Symposium on Wearable Computers*. IEEE Comput. Soc, Atlanta, GA, USA, 193–194. DOI:http://dx.doi.org/10.1109/ISWC.2000.888497

[248] Argha Sen, Anirban Das, Swadhin Pradhan, and Sandip Chakraborty. 2024. Continuous Multi-user Activity Tracking via Room-Scale mmWave Sensing. In *2024 23rd ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 163–175. DOI:http://dx.doi.org/10.1109/IPSN61024.2024.00018

[249] Arindam Sengupta, Feng Jin, Renyuan Zhang, and Siyang Cao. 2020. mm-Pose: Real-Time Human Skeletal Posture Estimation Using mmWave Radars and CNNs. *IEEE Sensors Journal* 20, 17 (Sept. 2020), 10032–10044. DOI:http://dx.doi.org/10.1109/JSEN.2020.2991741 Conference Name: IEEE Sensors Journal.

[250] Sensel. 2021. Sensel Morph. (2021). https://morph.sensel.com/

[251] Matthias Seuter, Alexandra Pollock, Gernot Bauer, and Christian Kray. 2020. Recognizing Running Movement Changes with Quaternions on a Sports Watch. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 4, Article 151 (Dec. 2020), 18 pages. DOI:http://dx.doi.org/10.1145/3432197

[252] Shuai Shao, Yu Guan, Bing Zhai, Paolo Missier, and Thomas Plötz. 2023. ConvBoost: Boosting ConvNets for Sensor-based Activity Recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 2 (June 2023), 75:1–75:21. DOI:http://dx.doi.org/10.1145/3596234

[253] Sheng Shen, Mahanth Gowda, and Romit Roy Choudhury. 2018. Closing the Gaps in Inertial Motion Tracking. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom '18)*. Association for Computing Machinery, New York, NY, USA, 429–444. DOI:http://dx.doi.org/10.1145/3241539.3241582

[254] Sheng Shen, He Wang, and Romit Roy Choudhury. 2016. I am a Smartwatch and I can Track my User's Arm. In *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '16)*. Association for Computing Machinery, New York, NY, USA, 85–96. DOI:http://dx.doi.org/10.1145/2906388.2906407

[255] Shixin (Cindy) Shen and Ronald A. House. 2017. Hand-arm vibration syndrome. *Canadian Family Physician* 63, 3 (March 2017), 206–210. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5349719/

[256] Vivian Shen, James Spann, and Chris Harrison. 2021. FarOut Touch: Extending the Range of ad hoc Touch Sensing with Depth Cameras. In *Symposium on Spatial User Interaction*. ACM, Virtual Event USA, 1–12. DOI:http://dx.doi.org/10.1145/3485279.3485281

[257] Xiyuan Shen, Chun Yu, Xutong Wang, Chen Liang, Haozhan Chen, and Yuanchun Shi. 2024. MouseRing: Always-available Touchpad Interaction with IMU Rings. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–19. DOI:http://dx.doi.org/10.1145/3613904.3642225

[258] Chunyuan Shi, Le Qi, Dapeng Yang, Jingdong Zhao, and Hong Liu. 2019. A Novel Method of Combining Computer Vision, Eye-Tracking, EMG, and IMU to Control Dexterous Prosthetic Hand. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, Dali, China, 2614–2618. DOI:http://dx.doi.org/10.1109/ROBIO49542.2019.8961582

[259] Yilei Shi, Haimo Zhang, Jiashuo Cao, and Suranga Nanayakkara. 2020a. VersaTouch: A Versatile Plug-and-Play System that Enables Touch Interactions on Everyday Passive Surfaces. In *Proceedings of the Augmented Humans International Conference (AHs '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. DOI:http://dx.doi.org/10.1145/3384657.3384778

[260] Yilei Shi, Haimo Zhang, Kaixing Zhao, Jiashuo Cao, Mengmeng Sun, and Suranga Nanayakkara. 2020b. Ready, Steady, Touch!: Sensing Physical Contact with a Finger-Mounted IMU. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 2 (June 2020), 1–25. DOI:http://dx.doi.org/10.1145/3397309

[261] Tsukasa Shiota, Motohiro Takagi, Kaori Kumagai, Hitoshi Seshimo, and Yushi Aono. 2024. Egocentric Action Recognition by Capturing Hand-Object Contact and Object State. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Waikoloa, HI, USA, 6527–6537. DOI:http://dx.doi.org/10.1109/WACV57701.2024.00641

[262] Takaaki Shiratori, Hyun Soo Park, Leonid Sigal, Yaser Sheikh, and Jessica K. Hodgins. 2011. Motion Capture from Body-Mounted Cameras. In *ACM SIGGRAPH 2011 Papers (SIGGRAPH '11)*. Association for Computing Machinery, New York, NY, USA, Article 31, 10 pages. DOI:http://dx.doi.org/10.1145/1964921.1964926

[263] Garth Shoemaker, Anthony Tang, and Kellogg S. Booth. 2007. Shadow reaching: a new perspective on interaction for large displays. *Proceedings of the 20th annual ACM symposium on User interface software and technology* (Oct. 2007), 53–56. DOI:http://dx.doi.org/10.1145/1294211.1294221 Conference Name: UIST07: The 20th Annual ACM Symposium on User Interface Software and Technology ISBN: 9781595936790 Place: Newport Rhode Island USA Publisher: ACM.

[264] Nabeel Siddiqui and Rosa H. M. Chan. 2020. Multimodal hand gesture recognition using single IMU and acoustic measurements at wrist. *PLOS ONE* 15, 1 (01 2020), 1–12. DOI:http://dx.doi.org/10.1371/journal.pone.0227039

[265] Yongkun Song, Tian Jin, Yongpeng Dai, Yongping Song, and Xiaolong Zhou. 2021. Through-Wall Human Pose Reconstruction via UWB MIMO Radar and 3D CNN. *Remote Sensing* 13, 2 (Jan. 2021), 241. DOI:http://dx.doi.org/10.3390/rs13020241

[266] Elizabeth Spencer. 2020. The Pocket: A Hidden History of Women's Lives, 1660-1900. *Cultural and Social History* 17, 4 (Aug. 2020), 570–572. DOI:http://dx.doi.org/10.1080/14780038.2020.1810955 Publisher: Routledge _eprint: https://doi.org/10.1080/14780038.2020.1810955.

[267] Srinath Sridhar, Anders Markussen, Antti Oulasvirta, Christian Theobalt, and Sebastian Boring. 2017. WatchSense: On- and Above-Skin Input Sensing through a Wearable Depth Sensor. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, New York, NY, USA, 3891–3902. DOI:http://dx.doi.org/10.1145/3025453.3026005

[268] T. Starner, J. Auxier, D. Ashbrook, and M. Gandy. 2000. The gesture pendant: a self-illuminating, wearable, infrared computer vision system for home automation control and medical monitoring. In *Digest of Papers. Fourth International Symposium on Wearable Computers*. 87–94. DOI:http://dx.doi.org/10.1109/ISWC.2000.888469

[269] STMicroelectronics. 2020. STMicroelectronics Introduces World's First All-in-One, Multi-Zone, Direct Time-of-Flight Module. (2020). https://newsroom.st.com/media-center/press-item.html/p4281.html

[270] STMicroelectronics. 2022a. 2nd-generation multi-zone direct Time-of-Flight sensor from STMicroelectronics uses less energy and can range 2X as far as existing products. (June 2022). https://www.globenewswire.com/news-release/2022/06/09/2459735/0/en/2nd-generation-multi-zone-direct-Time-of-Flight-sensor-from-STMicroelectronics-uses-less-en html

[271] STMicroelectronics. 2022b. VL53L8CX - Low-power high-performance 8x8 multizone Time-of-Flight sensor (ToF) - STMicroelectronics. (2022). https://www.st.com/en/imaging-and-photonics-solutions/vl53l8cx.html

[272] STMicroelectronics. 2026. STMicroelectronics Brings Always-On Vision to Next-Generation Personal Electronics with New Ultralow-Power Image Sensors. Press release, ST Newsroom. (April 2026). https://newsroom.st.com/media-center/press-item.html/p4772.html Accessed: 2026-09-15.

[273] Paul Streli, Jiaxi Jiang, Juliete Rossie, and Christian Holz. 2023. Structured Light Speckle: Joint Ego-Centric Depth Estimation and Low-Latency Contact Detection via Remote Vibrometry. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. ACM, San Francisco CA USA, 1–12. DOI:http://dx.doi.org/10.1145/3586183.3606749

[274] Paul Streli, Mark Richardson, Fadi Botros, Shugao Ma, Robert Wang, and Christian Holz. 2024. TouchInsight: Uncertainty-aware Rapid Touch and Text Input for Mixed Reality from Egocentric Vision. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. ACM, Pittsburgh PA USA, 1–16. DOI:http://dx.doi.org/10.1145/3654777.3676330

[275] Joshua Strickon and Joseph Paradiso. 1998. Tracking hands above large interactive surfaces with a low-cost scanning laser rangefinder. In *CHI 98 Conference Summary on Human Factors in Computing Systems (CHI '98)*. Association for Computing Machinery, New York, NY, USA, 231–232. DOI:http://dx.doi.org/10.1145/286498.286719

[276] David Strömbäck, Sangxia Huang, and Valentin Radu. 2020. MM-Fit: Multimodal Deep Learning for Automatic Exercise Logging across Sensing Devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 4 (Dec. 2020), 168:1–168:22. DOI:http://dx.doi.org/10.1145/3432701

[277] Naoki Sugita, Daisuke Iwai, and Kosuke Sato. 2008. Touch sensing by image analysis of fingernail. In *2008 SICE Annual Conference*. 1520–1525. DOI:http://dx.doi.org/10.1109/SICE.2008.4654901

[278] Y. Sugiura, F. Nakamura, W. Kawai, T. Kikuchi, and M. Sugimoto. 2017. Behind the palm: Hand gesture recognition through measuring skin deformation on back of hand by using optical sensors. In *2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*. 1082–1087. DOI:http://dx.doi.org/10.23919/SICE.2017.8105457

[279] Wei Sun, Franklin Mingzhe Li, Congshu Huang, Zhenyu Lei, Benjamin Steeper, Songyun Tao, Feng Tian, and Cheng Zhang. 2021. ThumbTrak: Recognizing Micro-finger Poses Using a Ring with Proximity Sensing. In *Proceedings of the 23rd International Conference on Mobile Human-Computer Interaction (MobileHCI '21)*. Association for Computing Machinery, New York, NY, USA, 1–9. DOI:http://dx.doi.org/10.1145/3447526.3472060

[280] Arvin Tashakori, Zenan Jiang, Amir Servati, Saeid Soltanian, Harishkumar Narayana, Katherine Le, Caroline Nakayama, Chieh-ling Yang, Z. Jane Wang, Janice J. Eng, and Peyman Servati. 2024. Capturing complex hand movements and object interactions using machine learning-powered stretchable smart textile gloves. *Nature Machine Intelligence* 6, 1 (Jan. 2024), 106–118. DOI:http://dx.doi.org/10.1038/s42256-023-00780-9

[281] tesych. 2020. About Azure Kinect DK. (Nov. 2020). https://docs.microsoft.com/en-us/azure/kinect-dk/about-azure-kinect-dk

[282] Marian Theiss, Philipp M. Scholl, and Kristof Van Laerhoven. 2016. Predicting Grasps with a Wearable Inertial and EMG Sensing Unit for Low-Power Detection of In-Hand Objects. In *Proceedings of the 7th Augmented Human International Conference 2016 (AH '16)*. Association for Computing Machinery, New York, NY, USA, 1–8. DOI:http://dx.doi.org/10.1145/2875194.2875207

[283] Joseph Thomas. 2013. A Camera Based Virtual Keyboard with Touch Detection by Shadow Analysis. https://www.semanticscholar.org/paper/A-Camera-Based-Virtual-Keyboard-with-Touch-by-Thomas/fa13b49719fd6868ace04445237ade1fc600e42b

[284] Hoang Truong, Shuo Zhang, Ufuk Muncuk, Phuc Nguyen, Nam Bui, Anh Nguyen, Qin Lv, Kaushik Chowdhury, Thang Dinh, and Tam Vu. 2018. CapBand: Battery-free Successive Capacitance Sensing Wristband for Hand Gesture Recognition. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*. ACM, Shenzhen China, 54–67. DOI:http://dx.doi.org/10.1145/3274783.3274854

[285] Pavan Kumar Anasosalu Vasu, James Gabriel, Jeff Zhu, Oncel Tuzel, and Anurag Ranjan. 2023. FastViT: A Fast Hybrid Vision Transformer using Structural Reparameterization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

[286] Radu-Daniel Vatavu and Ionuţ Alexandru Zaiţi. 2013. Automatic recognition of object size and shape via user-dependent measurements of the grasping hand. *International Journal of Human-Computer Studies* 71, 5 (2013), 590–607. DOI:http://dx.doi.org/https://doi.org/10.1016/j.ijhcs.2013.01.002

[287] Vicon. 2020. Vicon | Award Winning Motion Capture Systems. (2020). https://www.vicon.com

[288] Boga Vishal and K Deepak Lawrence. 2017. Paper piano — Shadow analysis based touch interaction. In *2017 2nd International Conference on Man and Machine Interfacing (MAMI)*. 1–6. DOI:http://dx.doi.org/10.1109/MAMI.2017.8307890

[289] Daniel Vlasic, Rolf Adelsberger, Giovanni Vannucci, John Barnwell, Markus Gross, Wojciech Matusik, and Jovan Popović. 2007. Practical motion capture in everyday surroundings. *ACM Transactions on Graphics* 26, 3 (July 2007), 35. DOI:http://dx.doi.org/10.1145/1276377.1276421

[290] Timo von Marcard, Bodo Rosenhahn, Michael J. Black, and Gerard Pons-Moll. 2017. Sparse Inertial Poser: Automatic 3D Human Pose Estimation from Sparse IMUs. (March 2017). http://arxiv.org/abs/1703.08014 arXiv:1703.08014 [cs].

[291] Richard Voyles, Jaewook Bae, and Roy Godzdanker. 2008. The gestural joystick and the efficacy of the path tortuosity metric for human/robot interaction. In *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems (PerMIS '08)*. Association for Computing Machinery, New York, NY, USA, 91–97. DOI:http://dx.doi.org/10.1145/1774674.1774689

[292] Anandghan Waghmare, Youssef Ben Taleb, Ishan Chatterjee, Arjun Narendra, and Shwetak Patel. 2023a. Z-Ring: Single-Point Bio-Impedance Sensing for Gesture, Touch, Object and User Recognition. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–18. DOI:http://dx.doi.org/10.1145/3544548.3581422

[293] Anandghan Waghmare, Roger Boldu, Eric Whitmire, and Wolf Kienzle. 2023b. OptiRing: Low-Resolution Optical Sensing for Subtle Thumb-to-Index Micro-Interactions. In *Proceedings of the 2023 ACM Symposium on Spatial User Interaction*. ACM, Sydney NSW Australia, 1–13. DOI:http://dx.doi.org/10.1145/3607822.3614538

[294] Naoki Wake, Atsushi Kanehira, Kazuhiro Sasabuchi, Jun Takamatsu, and Katsushi Ikeuchi. 2025. Open-Vocabulary Action Localization with Iterative Visual Prompting. *IEEE Access* 13 (2025), 56908–56917. DOI:http://dx.doi.org/10.1109/ACCESS.2025.3555167 arXiv:2408.17422 [cs.CV].

[295] Chuyu Wang, Jian Liu, Yingying Chen, Lei Xie, Hong Bo Liu, and Sanclu Lu. 2018. RF-Kinect: A Wearable RFID-based Approach Towards 3D Body Movement Tracking. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (March 2018), 1–28. DOI:http://dx.doi.org/10.1145/3191773

[296] Cheng-Yao Wang, Min-Chieh Hsiu, Po-Tsung Chiu, Chiao-Hui Chang, Liwei Chan, Bing-Yu Chen, and Mike Y. Chen. 2015a. PalmGesture: Using Palms as Gesture Interfaces for Eyes-free Input. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '15)*. Association for Computing Machinery, New York, NY, USA, 217–226. DOI:http://dx.doi.org/10.1145/2785830.2785885

[297] Edward J. Wang, Tien-Jui Lee, Alex Mariakakis, Mayank Goel, Sidhant Gupta, and Shwetak N. Patel. 2015b. MagnifiSense: inferring device interaction using wrist-worn passive magneto-inductive sensors. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 15–26. DOI:http://dx.doi.org/10.1145/2750858.2804271

[298] Liang Wang, Tao Gu, Xianping Tao, and Jian Lu. 2017. Toward a Wearable RFID System for Real-Time Activity Recognition Using Radio Patterns. *IEEE Transactions on Mobile Computing* 16, 1 (Jan. 2017), 228–242. DOI:http://dx.doi.org/10.1109/TMC.2016.2538230

[299] Wei Wang, Alex X. Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and Modeling of WiFi Signal Based Human Activity Recognition. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15)*. Association for Computing Machinery, New York, NY, USA, 65–76. DOI:http://dx.doi.org/10.1145/2789168.2790093

[300] Qingxin Wei, Jiaming Huang, Yi Gao, and Wei Dong. 2025. One Model to Fit Them All: Universal IMU-based Human Activity Recognition with LLM-assisted Cross-dataset Representation. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 9, 3 (Sept. 2025), 139:1–139:22. DOI:http://dx.doi.org/10.1145/3749509

[301] Wenchuan Wei, Keiko Kurita, Jilong Kuang, and Alex Gao. 2021. Real-Time 3D Arm Motion Tracking Using the 6-axis IMU Sensor of a Smartwatch. In *2021 IEEE 17th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*. 1–4. DOI:http://dx.doi.org/10.1109/BSN51625.2021.9507012

[302] Hongyi Wen, Julian Ramos Rojas, and Anind K. Dey. 2016. Serendipity: Finger Gesture Recognition using an Off-the-Shelf Smartwatch. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. Association for Computing Machinery, New York, NY, USA, 3847–3851. DOI:http://dx.doi.org/10.1145/2858036.2858466

[303] Jason Wiese, T. Scott Saponas, and A.J. Bernheim Brush. 2013. Phoneprioception: enabling mobile phones to infer where they are kept. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, Paris France, 2157–2166. DOI:http://dx.doi.org/10.1145/2470654.2481296

[304] Wikipedia. 2022a. List of best-selling video games. (Aug. 2022). https://en.wikipedia.org/w/index.php?title=List_of_best-selling_video_games&oldid=1104319901 Page Version ID: 1104319901.

[305] Wikipedia. 2022b. List of UWB-enabled mobile devices. (Sept. 2022). https://en.wikipedia.org/w/index.php?title=List_of_UWB-enabled_mobile_devices&oldid=1109918576 Page Version ID: 1109918576.

[306] Wikipedia. 2025. Lux. (April 2025). https://en.wikipedia.org/w/index.php?title=Lux&oldid=1284696029 Page Version ID: 1284696029.

[307] Mariusz P. Wilk, Michael Walsh, and Brendan O'Flynn. 2021. Multimodal Sensor Fusion for Low-Power Wearable Human Motion Tracking Systems in Sports Applications. *IEEE Sensors Journal* 21, 4 (Feb. 2021), 5195–5212. DOI:http://dx.doi.org/10.1109/JSEN.2020.3030779 Conference Name: IEEE Sensors Journal.

[308] Andrew D. Wilson. 2005. PlayAnywhere: a compact interactive tabletop projection-vision system. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*. ACM, Seattle WA USA, 83–92. DOI:http://dx.doi.org/10.1145/1095034.1095047

[309] Andrew D. Wilson. 2010. Using a depth camera as a touch sensor. In *ACM International Conference on Interactive Tabletops and Surfaces*. ACM, Saarbrücken Germany, 69–72. DOI:http://dx.doi.org/10.1145/1936652.1936665

[310] Raphael Wimmer. 2010. Grasp sensing for human-computer interaction. In *Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction (TEI '11)*. Association for Computing Machinery, New York, NY, USA, 221–228. DOI:http://dx.doi.org/10.1145/1935701.1935745

[311] 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*. ACM, Cambridge United Kingdom, 359–362. DOI:http://dx.doi.org/10.1145/1517664.1517736

[312] Frieder Wittmann, Olivier Lambercy, and Roger Gassert. 2019. Magnetometer-Based Drift Correction During Rest in IMU Arm Motion Tracking. *Sensors* 19, 6 (Jan. 2019), 1312. DOI:http://dx.doi.org/10.3390/s19061312

[313] Erwin Wu, Ye Yuan, Hui-Shyong Yeo, Aaron Quigley, Hideki Koike, and Kris M. Kitani. 2020b. 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 (UIST '20)*. Association for Computing Machinery, New York, NY, USA, 1147–1160. DOI:http://dx.doi.org/10.1145/3379337.3415897

[314] Jason Wu, Chris Harrison, Jeffrey P. Bigham, and Gierad Laput. 2020a. Automated Class Discovery and One-Shot Interactions for Acoustic Activity Recognition. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–14. DOI:http://dx.doi.org/10.1145/3313831.3376875

[315] Ziyi Xia, Xincheng Huang, Sidney S Fels, and Robert Xiao. 2025. HaloTouch: Using IR Multi-Path Interference to Support Touch Interactions with General Surfaces. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 548, 17 pages. DOI:http://dx.doi.org/10.1145/3706598.3714179

[316] Robert Xiao, Scott Hudson, and Chris Harrison. 2016. DIRECT: Making Touch Tracking on Ordinary Surfaces Practical with Hybrid Depth-Infrared Sensing. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces*. ACM, Niagara Falls Ontario Canada, 85–94. DOI:http://dx.doi.org/10.1145/2992154.2992173

[317] Robert Xiao, Scott Hudson, and Chris Harrison. 2017. Supporting Responsive Cohabitation Between Virtual Interfaces and Physical Objects on Everyday Surfaces. *Proc. ACM Hum.-Comput. Interact.* 1, EICS, Article 12 (June 2017), 17 pages. DOI:http://dx.doi.org/10.1145/3095814

[318] Robert Xiao, Julia Schwarz, Nick Throm, Andrew D. Wilson, and Hrvoje Benko. 2018. MRTouch: Adding Touch Input to Head-Mounted Mixed Reality. *IEEE Transactions on Visualization and Computer Graphics* 24, 4 (April 2018), 1653–1660. DOI:http://dx.doi.org/10.1109/TVCG.2018.2794222

[319] Xuesu Xiao and Shuayb Zarar. 2018. A Wearable System for Articulated Human Pose Tracking Under Uncertainty of Sensor Placement. In *2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics (Biorob)*. 1144–1150. DOI:http://dx.doi.org/10.1109/BIOROB.2018.8487858 ISSN: 2155-1782.

[320] Yingjing Xiao, ZhiChao Huang, and Yang Gao. 2025. From Wrist to Finger: Hand Pose Tracking Using Ring-Watch Wearables. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems (CHI EA '25)*. Association for Computing Machinery, New York, NY, USA, 1–7. DOI:http://dx.doi.org/10.1145/3706599.3720220

[321] Zhen Gang Xiao and Carlo Menon. 2017. Counting Grasping Action Using Force Myography: An Exploratory Study With Healthy Individuals. *JMIR rehabilitation and assistive technologies* 4, 1 (May 2017), e5. DOI:http://dx.doi.org/10.2196/rehab.6901 Number: 1.

[322] Xsens. 2020. Xsens Motion Capture. (2020). https://www.xsens.com/motion-capture

[323] Chao Xu, Parth H. Pathak, and Prasant Mohapatra. 2015. Finger-Writing with Smartwatch: A Case for Finger and Hand Gesture Recognition Using Smartwatch. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications (HotMobile '15)*. Association for Computing Machinery, New York, NY, USA, 9–14. DOI:http://dx.doi.org/10.1145/2699343.2699350

[324] Chenhan Xu, Bing Zhou, Gurunandan Krishnan, and Shree Nayar. 2023. AO-Finger: Hands-free Fine-grained Finger Gesture Recognition via Acoustic-Optic Sensor Fusing. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 306, 14 pages. DOI:http://dx.doi.org/10.1145/3544548.3581264

[325] Vasco Xu, Brian Chen, Eric J. Gonzalez, Andrea Colaço, Henry Hoffmann, Mar Gonzalez-Franco, and Karan Ahuja. 2026. SurfaceXR: Fusing Smartwatch IMUs and Egocentric Hand Pose for Seamless Surface Interactions. *IEEE Transactions on Visualization and Computer Graphics* 32, 5 (May 2026), 3745–3753. DOI:http://dx.doi.org/10.1109/TVCG.2026.3679136

[326] Xuhai Xu, Jun Gong, Carolina Brum, Lilian Liang, Bongsoo Suh, Shivam Kumar Gupta, Yash Agarwal, Laurence Lindsey, Runchang Kang, Behrooz Shahsavari, Tu Nguyen, Heriberto Nieto, Scott E Hudson, Charlie Maalouf, Jax Seyed Mousavi, and Gierad Laput. 2022. Enabling

Hand Gesture Customization on Wrist-Worn Devices. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–19. DOI:http://dx.doi.org/10.1145/3491102.3501904

[327] Yuki Yamato, Yutaro Suzuki, Kodai Sekimori, Buntarou Shizuki, and Shin Takahashi. 2020. Hand Gesture Interaction with a Low-Resolution Infrared Image Sensor on an Inner Wrist. (2020), 5.

[328] Chao Yang, Lingxiao Wang, Xuyu Wang, and Shiwen Mao. 2022. Environment Adaptive RFID-Based 3D Human Pose Tracking With a Meta-Learning Approach. *IEEE Journal of Radio Frequency Identification* 6 (2022), 413–425. DOI:http://dx.doi.org/10.1109/JRFID.2022.3140256 Conference Name: IEEE Journal of Radio Frequency Identification.

[329] Koji Yatani and Khai N. Truong. 2012. BodyScope: a wearable acoustic sensor for activity recognition. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12)*. Association for Computing Machinery, New York, NY, USA, 341–350. DOI:http://dx.doi.org/10.1145/2370216.2370269

[330] Hui-Shyong Yeo, Erwin Wu, Juyoung Lee, Aaron Quigley, and Hideki Koike. 2019. Opisthenar: Hand Poses and Finger Tapping Recognition by Observing Back of Hand Using Embedded Wrist Camera. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New York, NY, USA, 963–971. DOI:http://dx.doi.org/10.1145/3332165.3347867

[331] Taeyoung Yeon, Vasco Xu, Henry Hoffmann, and Karan Ahuja. 2025. WatchHAR: Real-time On-device Human Activity Recognition System for Smartwatches. In *Proceedings of the 27th International Conference on Multimodal Interaction (ICMI '25)*. Association for Computing Machinery, New York, NY, USA, 387–394. DOI:http://dx.doi.org/10.1145/3716553.3750775

[332] Xinyu Yi, Yuxiao Zhou, Vladislav Golyanik, Marc Habermann, Soshi Shimada, Christian Theobalt, and Feng Xu. 2022. Physical Inertial Poser (PIP): Physics-aware Real-time Human Motion Tracking from Sparse Inertial Sensors. (June 2022), 15.

[333] Xinyu Yi, Yuxiao Zhou, and Feng Xu. 2021. TransPose: real-time 3D human translation and pose estimation with six inertial sensors. *ACM Transactions on Graphics* 40, 4 (July 2021), 86:1–86:13. DOI:http://dx.doi.org/10.1145/3450626.3459786

[334] Yafeng Yin, Lei Xie, Tao Gu, Yijia Lu, and Sanglu Lu. 2019. AirContour: Building Contour-Based Model for In-Air Writing Gesture Recognition. *ACM Trans. Sen. Netw.* 15, 4, Article 44 (Oct. 2019), 25 pages. DOI:http://dx.doi.org/10.1145/3343855

[335] Yole Group. 2023. The MEMS Industry: Looking Back at the Last 20 Years of Innovation and Growth. Yole Group Strategy Insights. (Sept. 2023). https://www.yolegroup.com/strategy-insights/mems-industry-looking-back-at-the-last-20-years-of-innovation-and-growth/ Accessed: 2026-09-15.

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

[337] Alexander David Young. 2010. Use of Body Model Constraints to Improve Accuracy of Inertial Motion Capture. In *2010 International Conference on Body Sensor Networks*. 180–186. DOI:http://dx.doi.org/10.1109/BSN.2010.30 ISSN: 2376-8894.

[338] Tianhong Catherine Yu, Guilin Hu, Ruidong Zhang, Hyunchul Lim, Saif Mahmud, Chi-Jung Lee, Ke Li, Devansh Agarwal, Shuyang Nie, Jinseok Oh, François Guimbretière, and Cheng Zhang. 2024a. Ring-a-Pose: A Ring for Continuous Hand Pose Tracking. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 4 (Nov. 2024), 189:1–189:30. DOI:http://dx.doi.org/10.1145/3699741

[339] Tianhong Catherine Yu, Manru Mary Zhang, Peter He, Chi-Jung Lee, Cassidy Cheesman, Saif Mahmud, Ruidong Zhang, Francois Guimbretiere, and Cheng Zhang. 2024b. SeamPose: Repurposing Seams as Capacitive Sensors in a Shirt for Upper-Body Pose Tracking. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24)*. Association for Computing Machinery, New York, NY, USA, 1–13. DOI: http://dx.doi.org/10.1145/3654777.3676341

[340] Tianhong Catherine Yu, Manru Mary Zhang, Luis Miguel Malenab, Chi-Jung Lee, Jacky Hao Jiang, Ruidong Zhang, François Guimbretière, and Cheng Zhang. 2025. SeamFit: Towards Practical Smart Clothing for Automatic Exercise Logging. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 9, 1 (March 2025), 24:1–24:22. DOI: http://dx.doi.org/10.1145/3712287

[341] IonuŢ-Alexandru ZaiŢi, Radu-Daniel Vatavu, and Ştefan-Gheorghe Pentiuc. 2013. Exploring Hand Posture for Smart Mobile Devices. In *Human Factors in Computing and Informatics*, Andreas Holzinger, Martina Ziefle, Martin Hitz, and Matjaž Debevc (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 721–731.

[342] Mehrshad Zandigohar, Mo Han, Mohammadreza Sharif, Sezen Yağmur Günay, Mariusz P. Furmanek, Mathew Yarossi, Paolo Bonato, Cagdas Onal, Taşkın Padır, Deniz Erdoğmuş, and Gunar Schirner. 2024. Multimodal fusion of EMG and vision for human grasp intent inference in prosthetic hand control. *Frontiers in Robotics and AI* 11 (Feb. 2024). DOI:http://dx.doi.org/10.3389/frobt.2024.1312554 Publisher: Frontiers.

[343] Cheng Zhang, Qiuyue Xue, Anandghan Waghmare, Sumeet Jain, Yiming Pu, Sinan Hersek, Kent Lyons, Kenneth A. Cunefare, Omer T. Inan, and Gregory D. Abowd. 2017. SoundTrak: Continuous 3D Tracking of a Finger Using Active Acoustics. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 2 (June 2017), 1–25. DOI:http://dx.doi.org/10.1145/3090095

[344] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. 2020. MediaPipe Hands: On-device Real-time Hand Tracking. (June 2020). DOI:http://dx.doi.org/10.48550/arXiv.2006.10214 arXiv:2006.10214 [cs].

[345] Yang Zhang and Chris Harrison. 2015. Tomo: Wearable, Low-Cost Electrical Impedance Tomography for Hand Gesture Recognition. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, Charlotte NC USA, 167–173. DOI: http://dx.doi.org/10.1145/2807442.2807480

[346] Yang Zhang, Wolf Kienzle, Yanjun Ma, Shiu S. Ng, Hrvoje Benko, and Chris Harrison. 2019. ActiTouch: Robust Touch Detection for On-Skin AR/VR Interfaces. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New York, NY, USA, 1151–1159. DOI:http://dx.doi.org/10.1145/3332165.3347869

[347] Yang Zhang, Junhan Zhou, Gierad Laput, and Chris Harrison. 2016. SkinTrack: Using the Body as an Electrical Waveguide for Continuous Finger Tracking on the Skin. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, San Jose California USA, 1491–1503. DOI:http://dx.doi.org/10.1145/2858036.2858082

[348] Hongyang Zhao. 2018. MobiGesture_ Mobility-aware hand gesture recognition for healthcare. *Smart Health* (2018), 15.

[349] Yiming Zhao, Taein Kwon, Paul Streli, Marc Pollefeys, and Christian Holz. 2024. EgoPressure: A Dataset for Hand Pressure and Pose Estimation in Egocentric Vision. (2024). DOI: http://dx.doi.org/10.48550/ARXIV.2409.02224 Publisher: arXiv Version Number: 2.

[350] Han Zhou, Yi Gao, Xinyi Song, Wenxin Liu, and Wei Dong. 2020. LimbMotion: Decimeter-Level Limb Tracking for Wearable-Based Human-Computer Interaction. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 4, Article 161 (Sept. 2020), 24 pages. DOI:http://dx.doi.org/10.1145/3369836

[351] Huiyu Zhou, Thomas Stone, Huosheng Hu, and Nigel Harris. 2008. Use of multiple wearable inertial sensors in upper limb motion tracking. *Medical Engineering & Physics* 30, 1 (Jan. 2008), 123–133. DOI:http://dx.doi.org/10.1016/j.medengphy.2006.11.010

[352] Qian Zhou, Sarah Sykes, Sidney Fels, and Kenrick Kin. 2020. Gripmarks: Using Hand Grips to Transform In-Hand Objects into Mixed Reality Input. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–11. DOI:http://dx.doi.org/10.1145/3313831.3376313

[353] Shaghayegh Zihajehzadeh, Paul K. Yoon, Bong-Soo Kang, and Edward J. Park. 2015. UWB-Aided Inertial Motion Capture for Lower Body 3-D Dynamic Activity and Trajectory Tracking. *IEEE Transactions on Instrumentation and Measurement* 64, 12 (Dec. 2015), 3577–3587. DOI:http://dx.doi.org/10.1109/TIM.2015.2459532

[354] Thomas G Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, and Young Harvill. 1987. A HAND GESTURE INTERFACE DEVICE. (1987), 4.

[355] Quentin Zoppis, Sergi Pujades, Laurence Nigay, and François Bérard. 2025. Efficient Finger Model and Accurate Tracking for Hover-and-Touch, Mid-air and Microgesture Interaction. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology (UIST '25)*. Association for Computing Machinery, New York, NY, USA, 1–14. DOI:http://dx.doi.org/10.1145/3746059.3747701