

MeCap: Whole-Body Digitization for Low-Cost VR/AR Headsets

Karan Ahuja Chris Harrison Mayank Goel Robert Xiao
Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213
{kahuja, chris.harrison, mayank, brx}@cs.cmu.edu

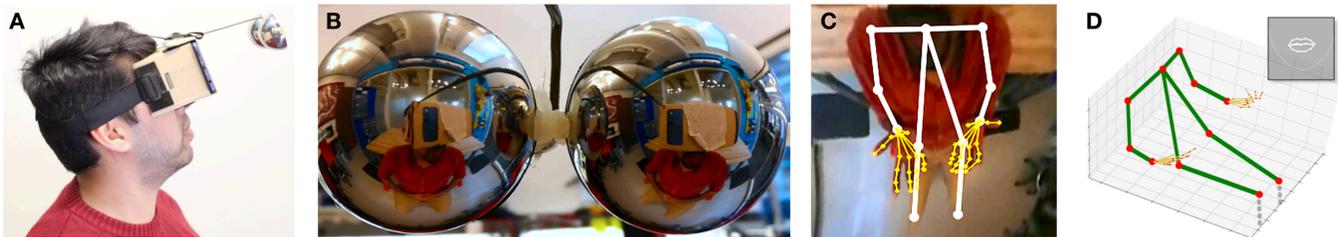


Figure 1. A) MeCap accessory attached to Google Cardboard. B) View of two mirrored spheres as seen by a smartphone's rear-facing camera. C) Unwrapped, virtual view looking down at a user's body, with hand and body keypoints overlaid. D) The 3D position of keypoints can be estimated through reverse raycasting stereo correspondence.

ABSTRACT

Low-cost, smartphone-powered VR/AR headsets are becoming more popular. These basic devices – little more than plastic or cardboard shells – lack advanced features, such as controllers for the hands, limiting their interactive capability. Moreover, even high-end consumer headsets lack the ability to track the body and face. For this reason, interactive experiences like social VR are underdeveloped. We introduce MeCap, which enables commodity VR headsets to be augmented with powerful motion capture (“MoCap”) and user-sensing capabilities at very low cost (under \$5). Using only a pair of hemi-spherical mirrors and the existing rear-facing camera of a smartphone, MeCap provides real-time estimates of a wearer's 3D body pose, hand pose, facial expression, physical appearance and surrounding environment – capabilities which are either absent in contemporary VR/AR systems or which require specialized hardware and controllers. We evaluate the accuracy of each of our tracking features, the results of which show imminent feasibility.

Author Keywords

Virtual Reality; Augmented Reality; Headset; Motion Capture; Hand Gestures; On-Body Sensing

INTRODUCTION

Low-cost virtual reality (VR) headsets, powered by a user's smartphone, have become increasingly popular since the introduction of Google Cardboard [14] in 2014. Since then,

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

UIST '19, October 20-23, 2019, New Orleans, LA, USA

© 2019 Association of Computing Machinery.

ISBN 978-1-4503-6816-2/19/10 \$15.00

<https://doi.org/10.1145/3332165.3347889>

multiple designs have emerged, such as Google's Daydream [15] and Samsung's Gear VR [41]. While the smartphone provides powerful compute and sensing capabilities, the interactive VR experience is typically limited to rudimentary physical controls on the headset itself.

More expensive headsets – such as the Oculus Rift [34], Sony PlayStation VR [44] and HTC Vive [21] – often bundle wireless, battery-powered hand controllers to boost interactive expressivity. These enable tracking of hand position and provide physical buttons for basic grips and actions. However, no consumer headset today provides tracking of the body or mouth – aspects that are clearly important in immersive virtual experiences, from full-body gaming [8] to social avatars [2]. The few prototype and research systems that do achieve body, hand and face tracking (reviewed later), tend to be expensive and often require external infrastructure, limiting their practicality and commercial feasibility.

In response, we developed a new, low-cost way to capture the body, hands, mouth and even apparel of a wearer from a VR/AR headset, requiring no external infrastructure or new sensors. We achieve this by using a smartphone's rear-facing camera to view two mirrored half-spheres, which are cantilevered roughly 15cm (~6”) in front of a headset (Figure 1A). This accessory, costing less than \$5 USD, offers a distorted but complete view of the wearer (Figure 1B). We process the two spherical images and render multiple, unwrapped, synthetic viewpoints (Figures 1C and 2). We can then take advantage of existing, state-of-the-art keypoint labeling algorithms to digitize the body and hands (Figure 1C, white and yellow lines). A single keypoint-labeled image does not innately provide depth, and so we use both unwrapped views for stereo correspondence to provide the missing *z*-axis data (Figure 1D and 8). We use additional processes for detecting hand pose (Figure 5 and 6), skin and apparel (Figure 4), and mouth state (Figure 12).

Taken together, these features bring the promise of expressive, fluid and controller-free input in VR/AR experiences closer to reality, especially in a low-cost manner that could be viable in consumer devices. Although future work remains on making the headset's industrial design more practical, the central premise should hold that a handful of wide-angle cameras operating in or around a headset (e.g. Figure 14B) could unlock significant interactive value, beyond what we see in contemporary VR/AR devices.

RELATED WORK

We are not the first to recognize the value of digitizing users' bodies and capturing personal appearance. Thus, we first review related work in motion capture, sensed both externally and by worn hardware. We then discuss research efforts to extend the capabilities of VR/AR headsets, especially low-cost, smartphone-based units.

External Motion Capture

There are two main approaches used for motion capture. The first is to instrument a person, most commonly retroreflective markers that can be captured by many high-speed infrared cameras [27], [51]. Such systems have been used extensively in the motion picture industry to capture high-quality body, hand and face pose. Popular commercial systems used in HCI research include Vicon [49] and OptiTrack [36].

There is also great interest in instrumentation-free approaches, which generally rely on computer vision techniques. Early efforts required depth cameras to robustly estimate pose, such as the original Microsoft Kinect sensor [33]. However, more recently, deep learning has demonstrated promising results using only RGB cameras [6], [28]. In VR settings, FaceVR [48] uses an external RGB+depth camera to capture the face and mouth pose and an internal IR camera (i.e., in the headset) to track the eyes. Researchers have also instrumented floors to track position and coarse user pose [3], [5]. Non-visual sensing modalities have also been considered for pose tracking, such as RF reflection [53] and capacitive sensing [55], but these have limited range and precision.

Worn Motion Capture

Being able to digitize user pose and input without having to rely on external infrastructure is an active area of research. This requires the use of worn sensors, which are generally specialized to capture an area of the body.

Most pervasive are systems that focus on digitizing the hands. Many technical approaches have been considered, including passive [3], [17] and active [29], [54] acoustic systems. Worn electrical approaches include electric field sensing [57] and impedance tomography [55]. Optical methods include infrared sensors for micro-gesture input [13] and motion sensors for wrist bending gestures [12]. Body-worn cameras have been used for tracking touch input on the skin [18], as well as hand segmentation and tracking [23], [25].

For whole-body capture, "suits" with distributed sensors are common. For example, XSens MVN [39], Tautges et al. [45]

and Ha et al. [16] used an array of IMUs on the body. Shiratori et al. [42] used 16 wide-angle body mounted cameras for motion reconstruction. There are also exoskeleton-style suits which use mechanical linkages, such as the METAmotion Gypsy [30].

Extending the Capability of AR/VR Headsets

More recently, there has been work looking to extend the sensing capabilities of AR/VR headsets. Olszski et al. [35] and Li et al. [26] used a camera mounted to the underside of a headset to capture mouth pose, while Thalmann et al. [47] added a depth camera to an AR headset for hand tracking. Closest to our work technically is EgoCap [38], which used a pair of add-on fisheye cameras cantilevered from a VR headset for body pose estimation. More closely related to the ethos of our work is EyeSpyVR [1], which used the existing front-facing camera of a smartphone, inside a low-cost VR headset, to capture eye movement and blinks.

While the above systems demonstrate body, hand *or* mouth tracking, none offer all three in a combined approach. Moreover, MeCap leverages the camera hardware already present in smartphones, making it extremely low cost compared to prior systems that require new sensing hardware, such as additional cameras. Existing headsets can be easily retrofitted with our MeCap attachment.

IMPLEMENTATION

Our proof-of-concept implementation required new hardware and software, which work together to enable hand, body, mouth, skin and apparel capture. We now describe the main components of our system.

Apparatus

We experimented with a wide variety of mirrored surfaces (e.g., mirrored glass, aluminized plastic), geometries (e.g., flat, parabolic) and arrangements (e.g., arranged vertically, staggered in distance) before arriving at our current design. Ultimately, we selected a pair of equidistant, horizontally arranged, 5.7 cm diameter, polished chrome steel hemispheres, costing \$1.78 each. These are cantilevered 15 cm out from a headset by a 1 mm hardened steel rod (Figure 1A), which strikes an important balance between visual occlusion, rigidity and weight.

Indeed, weight is a serious consideration, as the cantilever applies torque to the head, which can lead to discomfort. To minimize this, we place the spheres as close to the camera as possible (filling the view, while still permitting sharp focus) and also use hollowed hemispheres (0.6 mm wall diameter). The total mass cantilevered from the headset is 48 grams, making it less than the weight of a Leap Motion (52 grams), commonly affixed to the front of VR headsets. The total weight of the headset (with phone and MeCap attachment) is 295g, considerably less than e.g., HTC Vive at 470g or Oculus Go at 468g. Over the course of development, we added MeCap attachments to a Google Cardboard [14] (Figure 1A), VR Box headset [50], Holokit [20], and VuFine AR kit (Figure 14A), though it should be possible to add it to nearly any smartphone-based headset.



Figure 2. Equirectangular projection of image from sphere.

Unwarping and Viewpoint Selection

The spherical shape of the mirrors allows them to reflect essentially a full 360° panoramic image of the environment (i.e. a non-central catadioptric camera [31]), with blind spots behind the mirror itself and the reflection of the other mirror. These panoramic images are warped onto the apparent surface of the sphere when imaged by the phone's camera (Figure 1B), which precludes the use of many conventional computer vision algorithms. Thus, the first step is to unwarp both spherical mirror images.

Given the exact field-of-view of the camera and an undistorted camera image, we can compute the camera ray vector corresponding to any pixel in the scene (with the origin implied to lie at the camera's focal point). We use a manual calibration procedure to label the visual center of each sphere (which always reflects the center of the camera lens) and several points along the visible edge of the sphere. The physical center of the sphere lies along the visual center's pixel ray, and the rays to the visible edges are tangent to the sphere. Thus, in conjunction with the physical size of the sphere, we solve for the physical position of the sphere's center relative to the camera's focal point. For each visible point on the sphere, we can compute the camera ray vector to that point, and the resulting reflected ray, giving us a set of rays that start at various points on the sphere's surface and extend out towards the environment, each associated with a pixel in the original input image.

We can use these rays to generate novel viewpoints and camera projections from the input image. For instance, we can plot the input image pixels on a 2D grid according to the azimuth (X) and altitude (Y) of the corresponding sphere ray (relative to some fixed pole and meridian) and interpolate at grid points, producing an equirectangular projection of the input image (Figure 2) displaying a near full 360-by-180° map of the environment as seen in the spherical reflection. To obtain a perspective projection, we approximate the ball's surface as the focal point of a virtual camera (as the sphere rays do not actually converge on a single focal point), then plot and interpolate image pixels according to the angle at which the sphere rays enter the camera.

In order to speed up the perspective projection process, we texture-map the input image onto the surface of a spherical



Figure 3. Synthetic camera views MeCap uses to capture the hands (left) and body pose (right).

mesh, and then place a virtual camera inside the ball. With this, we can obtain a fairly accurate perspective projection simply by rendering the mesh with the desired virtual camera, exploiting the GPU to accelerate the ray computations. This enables real-time perspective projection rendering, and additionally allows us to easily select the virtual camera's direction and field-of-view to optimally capture the user's hands, body, or environment (two examples in Figure 3).

Body and Hand Tracking

Our tracking pipeline begins by taking the perspective unwrapped images from the left and right spheres and using OpenPose [6] for hand and body keypoint estimation on each image. The keypoint estimation provides chest, shoulder, elbow, wrist, hip and knee keypoints (11 in total), as well as 21 keypoints for each hand. At this stage in our pipeline, the effective resolution of the body/hand image is roughly 200×200 (i.e., the 1280×720 rear camera image is cropped into two 400×400 sphere ROIs, which are then unwrapped and cropped again to segment the wearer).

Stereo Correspondence Reverse Raycasting

The two sphere images provide two sets of body and hand keypoints from slightly different, but known viewpoints. This allows us to utilize stereo correspondence to estimate the 3D position of each keypoint [32]. Specifically, for any keypoint pair, we find the corresponding sphere rays consisting of the origin on the sphere surface together with the direction of the reflected ray. As these rays will rarely intersect perfectly due to measurement error, we instead locate the 3D point that lies closest to each ray.

Hand Gesture Recognition

For hand gesture recognition, we crop images of each hand from both unwrapped views using OpenPose's reported body keypoint of the hands. These patches are then fed into a convolutional neural network pre-initialized with VGG-16 ImageNet [43] weights (implemented in Keras [7] with a TensorFlow backend). A second parallel neural network is passed the hand keypoint direction unit vectors (using the wrist as the origin), and the outputs of both networks are concatenated into a final fully-connected layer. We train the aggregate network using a joint loss function, using the hand gesture as the output label.

Mouth State Recognition

For mouth state recognition, we first crop out an image of the mouth from the unwrapped view. This position is fixed as the

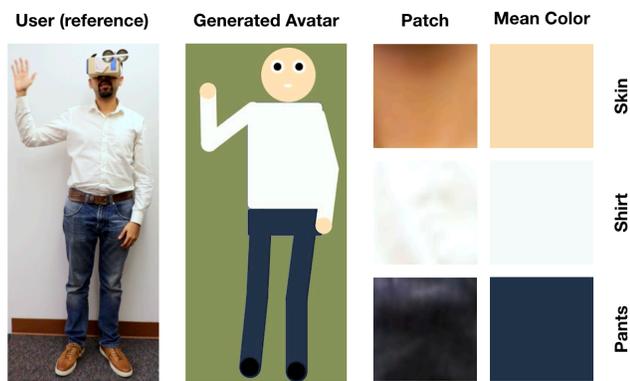


Figure 4. Example generated avatar following automatic skin and apparel detection.

VR headset is mounted to the face with a known geometry. We composite both images we have of the mouth (i.e., both spheres) and pass it through a convolutional neural network (VGG-16) pre-initialized with ImageNet weights. The neural network is then trained on one of five emoji-inspired mouth states: neutral, mouth open, smile, kissy face, and frown.

Skin and Apparel

For skin, we extract a small image patch from the user's neck and hands (as tracked by our earlier stages), as these are the least likely areas to be occluded by clothing. For apparel, we similarly extract image patches based on body keypoints (shirt is centroid between hips and chest; pants are extracted from left leg, between knee and hip). Once all patches have been extracted, we apply a gamma correction ($\gamma=1.5$) to account for loss of luminance. The phone was allowed to automatically white balance.

Overall, we found image patches to be highly variable in quality, with e.g., pattern asymmetries, partial occlusion and heavy folds leading to poor results, especially when attempting to tile the patch into a texture. For this reason, we also compute the median color for each patch, which offers a more regular and stable color reference for the skin, shirt and pants (Figure 4).

Performance

Our proof-of-concept system has two main compute elements. First is an Android service that captures video and streams it to a server using RTSP in the background. Second is a server hosted at our institution (Intel Core i7, NVidia 1080Ti GPU), which receives frames for processing and replies back with data for the aforementioned features.

Over Wi-Fi, average client-to-server video transmission latency was 108ms (at 1280x720). Once a frame is received, it takes 41 ms to unwarp both spherical views, 84 ms for OpenPose to provide body and hand keypoints, followed by 4 ms for our reverse raycasting stereo reconstruction. Using this data, our processes for hand gesture (8 ms), mouth state (3 ms), skin and apparel (1 ms) recognition run in parallel, taking another 8 ms. The server then relays all of this information back to the requesting client, taking another ~25 ms.

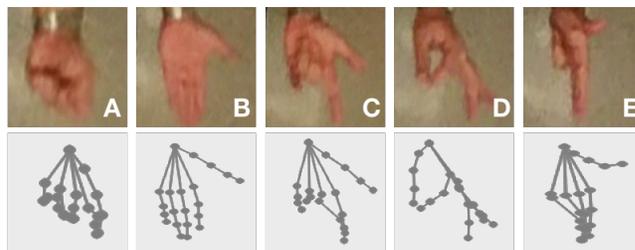


Figure 5. Top: One-handed gestures viewed from headset (cropped): A) fist, B) hand open, C) spiderman, D) okay and E) gun. Bottom: Detected hand keypoints.

In sum, the round-trip latency of our system is approximately 270 ms (SD = 32, min=252, max=308) and runs at a framerate of 12 Hz.

We note that almost half of our latency (~130 ms) is due to transmission of data to/from the server. Thus, in a commercial implementation, latency could be significantly reduced by processing everything locally. Indeed, current generation smartphones are on the edge of interactive feasibility with mobile implementations of PoseNet [37] running at 60 FPS using TensorFlow Lite [22]. Phone GPU/NN hardware is also quickly advancing; for example, the iPhone X is 50% faster than the previous generation using its new neural compute engine. In the next few years, latencies approaching 50 ms should be achievable.

EXAMPLE USES

To help convey the utility of MeCap, we now describe three illustrative uses which could be incorporated into full VR/AR applications to enhance immersion and input bandwidth. Please also see the Video Figure.

Social Avatars

Being able to express oneself is a fundamental component of social experiences. However, with today's AR/VR headsets, this typically means using controllers for hand tracking, with physical or virtual buttons to express gestures. Facebook's Venue is one such example, with facial gestures controlled with physical buttons on a hand controller [46]. In contrast, MeCap is controller-less, capturing actual body, hand and mouth pose, which could enable much more expressive AR/VR social experiences.

Additionally, contemporary social VR applications require users to design an avatar before use (e.g., VRChat [51], AltspaceVR [2], and Facebook Venue [46]). Changing

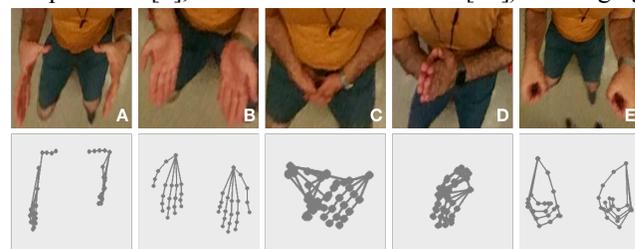


Figure 6. Top: Two-handed gestures viewed from headset: A) ball, B) book, C) relaxed, D) prayer and E) wheel. Bottom: Detected hand keypoints.

apparel, for instance, means re-entering an avatar design interface. Given the importance of expressing oneself through physical appearance in face-to-face contact, contemporary approaches would appear to be highly limiting. Again, in contrast, MeCap could allow for automatic updates of avatar clothing as an additional expressive channel. Figure 4 shows one example an automatically generated avatar.

Controller-Less Hand Input

Today’s VR/AR systems almost universally rely on physical controllers for tracking hand position and digitizing hand inputs. This is clearly a bottleneck in immersive 3D environments, which often feature graspable objects, tools and controls. Contemporary approaches also generally ignore leg actions, precluding e.g., kicking a ball. Controllers also increase cost and complexity of systems, and users must keep them charged for use.

As an illustration of MeCap’s potential to enable controller-less input in VR/AR, we built a hand gesture recognition demo that lets users summon different virtual objects, such as books and balls. Some items can be also manipulated, for example, a gun can be fired by pulling the trigger and a steering wheel can be turned. In total, our demo app supports five one-handed gestures (fist, hand open, spiderman, okay, and gun) and five bimodal gestures (ball, book, relaxed, prayer, and wheel), shown in Figures 5 and 6 and evaluated later.

Environment Map

In VR, where the environment is entirely virtual, it is relatively straightforward to correctly illuminate virtual objects. However, in AR, where virtual objects are placed into real environments, realistic lighting is challenging. This is chiefly because objects that face the user will be illuminated by sources to the side, above and behind the viewer. In conventional AR systems, this information is not known (and would have to be built up opportunistically, e.g., with SLAM [11]).

In MeCap, our spherical mirrors provide a near 360° view of a user’s surroundings, perfect for rendering an environment map. Indeed, mirrored spheres have long been used for this purpose in computational photography [9], [10], [19], [24], but have never been applied to AR. As a basic demo, using a VuFine AR headset (Figure 14A), we created an application that renders two balls in front of the user, one specular and one diffuse (Figure 7), based on real-time lighting.



Figure 7. View from front-facing camera (top) used to create a realistic environment map on two virtual spheres (bottom; specular and diffused reflection).

SPATIAL ACCURACY STUDY

Before evaluating high-level functionality, our first step was to quantify the low-level, 3D positional accuracy of our approach. Sources of error include optical distortion from our slightly aspherical mirrors, inaccuracy in estimating camera intrinsics, error in viewpoint selection, and innate resolution limitations from cropping a small region from an HD image.

To evaluate this, we setup two 5 cm checkerboard patterns, perpendicular to one another (Figure 8A), with the camera aperture 5 mm above and 5 mm behind the center axis of one plane. From this controlled viewpoint, the 3D coordinates of all intersections are known in real world units. This volume of space is roughly where the hands would operate. We then manually annotated all intersections (Figure 8B) and had MeCap compute the estimated 3D positions, which we can compare to actual positions.

Figure 8C shows the error result in the Y/Z plane (i.e., plane extending out from the camera) and the X/Y plane (i.e., plane



Figure 8. A) Orthogonal checkerboard setup, with headset mounted at top. B) View of checkerboard from headset with intersections denoted. C) 3D error in YZ and XY Planes.

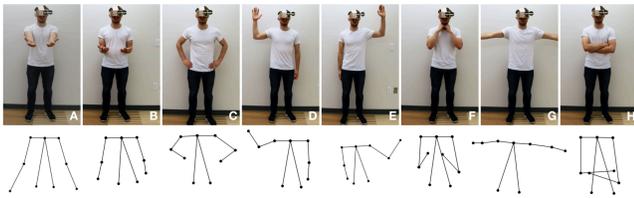


Figure 9. MeCap used eight body poses in its evaluation: A) arms stretched in front, B) arms held in front, C) hands on waist, D) left hand raised, E) right hand raised, F) hands on face, G) arms stretched horizontally, and H) arms crossed.

extending below and parallel the camera). Mean Euclidean distance error is 22.9 mm. As one would expect, the error grows as we move farther away from the center of the sphere.

USER STUDY

Our spatial accuracy study provided a controlled estimate of 3D positional accuracy. However, it provides little insight into how well our full-stack system works when worn. Thus, we ran a user study to evaluate our system’s performance when worn on real users. We recruited ten participants (3 female), who were paid \$20 for the one-hour study. Half of the participants completed the study seated, while the other half stood, which added variety and realism to our tests. The experimental apparatus is the same as that shown in Figure 1A. Testing each feature required a slightly different procedure, which is described in the sections below.

Sitting vs. Standing

We observed a slight performance benefit when participants were seated, as this provided a superior view of the legs. However, there was not a significant difference for the features we tested, and thus we combined results for our two sets of participants for all subsequent analyses.

Body Pose

We asked participants to perform eight body poses (Figure 9) in a random order: right hand raised, left hand raised, arms stretched horizontally, both hands on face, arms crossed, arms on hip, arms held in front, and arms stretched in front. While holding this position, an experimenter saved one frame of output for later analysis. These 80 frames (10 participants × 8 body poses) were then manually annotated with body keypoints (knee and above) and compared with the automatic OpenPose annotations to derive error statistics.

	Fist	Gun	Okay	Hand open	Spiderman
Fist	96.3%	0.3%	1.0%	0.0%	2.3%
Gun	0.3%	99.7%	0.0%	0.0%	0.0%
Okay	0.0%	3.0%	97.0%	0.0%	0.0%
Hand open	3.3%	0.0%	0.0%	93.3%	3.3%
Spiderman	1.0%	0.3%	1.0%	4.0%	93.7%

	Fist	Gun	Okay	Hand open	Spiderman
Fist	75.0%	0.0%	5.7%	14.0%	5.3%
Gun	5.7%	80.3%	12.0%	1.7%	0.3%
Okay	0.0%	9.7%	75.3%	10.3%	4.7%
Hand open	0.0%	0.0%	2.3%	96.3%	1.3%
Spiderman	1.0%	7.0%	7.3%	13.7%	71.0%

Figure 10. Confusion matrices for one-handed gestures cross-session (top) and cross-participant (bottom).

Body Keypoint	% Captured	Mean Normalized Misalignment Error
Chest	100.0	0.11
Shoulder	100.0	0.13
Elbow	100.0	0.20
Wrist	100.0	0.42
Hip	98.1	0.26
Knee	83.1	0.71

Table 1. Percentage of keypoints captured and normalized misalignment error for different body keypoints.

Statistics are normalized by participant body size, where 1.0 equals the participant’s shoulder width. For example, a normalized error of 0.2 means that keypoints were misaligned on average by one fifth of a shoulder width. We did not see any significant difference in performance between left and right keypoints, and so Table 1 collapses symmetric results.

Hand Gestures

We asked our participants to perform all gestures supported by our model (Figures 5 and 6), which included one-handed gestures (fist, hand open, okay, gun and spiderman) and two-handed gestures (holding a ball, holding a book, steering a wheel, prayer, and hands relaxed). These gesture sets were partially drawn from popular VR games. We collected data across 3 sessions, collecting 10 instances per gesture per session, which provided a total of 3000 instances (10 instances × 5 gestures × 2 gesture sets × 3 sessions × 10 participants).

We evaluated our gesture sets in two ways. First was a leave-one-session-out procedure, where all but one round of a participant’s data was used to train the hand gesture model, testing on the participant’s remaining hold-out session (all combinations). The second procedure was a leave-one-participant-out cross validation, in which the model was trained on all other users’ data, but was not exposed to any data from the wearer (simulating “out-of-the-box” accuracy).

For one-handed gestures, MeCap achieved an accuracy of 96.0% in a cross-session scenario and 79.6% in a cross-user scenario. For two handed gestures, the system achieved a cross-session accuracy of 97.7% and a cross-user accuracy of 80.2%. The confusion matrix for single and two-handed gestures is shown in Figures 10 and 11 respectively.

	Ball	Book	Relaxed	Prayer	Wheel
Ball	98.7%	0.0%	0.0%	0.0%	1.3%
Book	4.7%	92.7%	0.3%	0.3%	2.0%
Relaxed	0.0%	0.0%	99.3%	0.7%	0.0%
Prayer	0.0%	0.0%	0.0%	100.0%	0.0%
Wheel	2.3%	0.0%	0.0%	0.0%	97.7%

	Ball	Book	Relaxed	Prayer	Wheel
Ball	71.7%	10.0%	0.0%	0.0%	18.3%
Book	7.7%	74.3%	4.7%	0.7%	12.7%
Relaxed	0.3%	1.7%	96.0%	1.3%	0.7%
Prayer	0.0%	0.0%	0.0%	100.0%	0.0%
Wheel	9.3%	1.3%	0.3%	0.3%	88.7%

Figure 11. Confusion matrices for two-handed gestures cross-session (top) and cross-participant (bottom).



Figure 12. Mouth States: A) mouth closed, B) kissy face, C) mouth open, D) frown and E) smile.

Mouth State

MeCap supports five emoji-inspired mouth states: mouth open, mouth close, smile, frown and kissy face (Figure 12). To evaluate this feature’s accuracy, we collected 3 sessions of data across 10 participants, with each session capturing 10 images per mouth state. This resulted in a total of 1500 images (10 images × 5 states × 3 sessions × 10 participants). As before, we evaluated our system using a cross-user and cross-session leave-one-out cross validation procedure.

On these five mouth states, MeCap achieved a cross-session accuracy of 95.6% and cross-user accuracy of 64.5%. The confusion matrices are shown in Figure 13. At a high level, the results indicate that we do not have enough data to fine-tune a deep neural network over all mouth states, and that certain mouth states such as frown and kissy face do not generalize well across people. Hence, we performed a post-hoc analysis using two new classifiers, one to distinguish smile from no smile and the other to distinguish mouth open from mouth closed. The former achieves a cross-session accuracy of 100% and cross-user accuracy of 96.2%, whereas the latter achieves a cross-session accuracy of 100% and a cross-user accuracy of 98.7%.

Skin and Apparel Color

At the end of the study, we asked participants to evaluate the validity of MeCap’s skin and apparel color guesses. For skin color estimation, we asked participants to pick their own skin tone using a printed Fitzpatrick scale [40] (which has seven categories). We then asked them to place MeCap’s predicted skin color on the Fitzpatrick scale as well, and the difference was recorded. For 9 of 10 participants, the choices agreed, and for the 10th participant, the selection was off by one.

For shirt and pants, we showed participants MeCap’s extracted patches and colors. The participant then rated the quality of the extracted texture and color using a ten-point Likert scale, with one denoting “no resemblance” and ten for a “perfect match”. Overall, on shirts, MeCap achieved a mean Likert score of 6.4 (SD=4.6) for texture and 7.1 (SD=3.3) for color. For pants, the system achieved a mean Likert score of 7.0 (SD=3.7) on texture and 7.0 (SD=3.7) on color. In one case, a participant’s shirt had a large printed design without a dominant pattern or color, which our fixed-patch approach could not capture.

LIMITATIONS

There are several immediate and significant limitations in our current MeCap system that are worth noting. Foremost, by placing mirrors directly in front of the camera, the option for pass-through VR/AR is eliminated, as the view is occluded. To overcome this, we envision the hardware as a removable accessory. We also note that for low-cost AR

	Frown	Kissy Face	Mouth Closed	Mouth Open	Smile
Frown	87.0%	8.7%	4.0%	0.0%	0.3%
Kissy Face	3.7%	95.0%	1.3%	0.0%	0.0%
Mouth Close	1.0%	0.3%	98.7%	0.0%	0.0%
Mouth Open	0.0%	0.0%	0.0%	99.3%	0.7%
Smile	0.7%	0.0%	0.0%	1.3%	98.0%

	Frown	Kissy Face	Mouth Closed	Mouth Open	Smile
Frown	36.3%	28.3%	25.3%	2.3%	7.7%
Kissy Face	6.7%	72.0%	18.0%	2.0%	1.3%
Mouth Close	13.7%	12.7%	71.3%	0.0%	2.3%
Mouth Open	2.7%	9.3%	2.0%	80.0%	6.0%
Smile	4.0%	14.7%	10.7%	8.0%	62.7%

Figure 13. Confusion matrices for face states cross-session (top) and cross-participant (bottom).

headsets (e.g., VuFine; Figure 14A), MeCap would not block the user’s view, as a half-silvered mirror is used internally.

Another issue with our current design is that the legs are barely visible, and thus inaccurate to estimate. One possible solution is to move the mirrors farther out from the headset, but this reduces practicality and increases torque on the wearer’s head. Likewise, our current arrangement does not capture limbs operating behind the body (e.g., arms held behind back). Another limitation is reliance on ambient illumination and current low-light camera technologies. In darker environments, one potential solution would be to turn on the smartphone flash (which would reflect off the hemispheres) to provide some body illumination. We also note that the mirrors may get dirty from handling, which could degrade tracking performance. Rapid head movement also introduces motion blur, which might be mitigated with more sensitive and higher framerate cameras in the future.

To facilitate rapid development, and avoid compute limitations on mobile-class processors, we ran all computer vision processing on a network-connected desktop computer (akin to a cloud service). Of course, the low latency constraints in AR/VR interfaces likely precludes remote processing. Fortunately, smartphone manufacturers are increasingly including hardware accelerated deep learning capabilities, which should make local processing feasible in the coming years. However, one penalty we are unlikely to avoid any time soon is the energy cost of running the camera at 60 or more FPS.

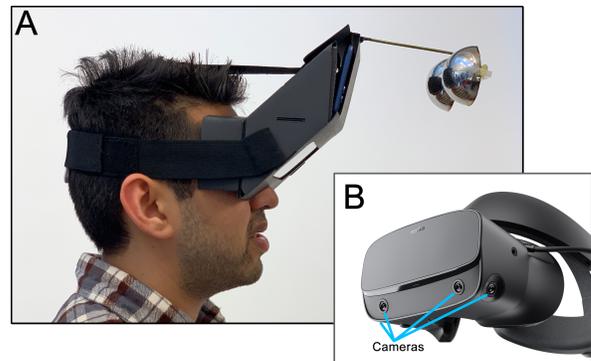


Figure 14. A) VuFine AR Kit with MeCap accessory attached. B) Commercial Oculus Rift S, which features four front facing cameras and could potentially run MeCap’s features.

Finally, we note the industrial design of our proof-of-concept device would have to significantly improve to provide the robustness and aesthetics required for consumer use. We envision a commercial design being similar to macro/wide-angle camera attachments for smartphones, containing two small spherical mirrors in a clear housing, with a lens for improved focus and field of view. It is also possible to integrate wide-angle cameras directly into AR/VR headsets to offer MeCap-like capabilities. For example, the recently announced Oculus Rift S [34] (Figure 14B) includes four cameras on the front of the headset, which could provide views of the wearer's body if sufficiently wide angle.

FUTURE WORK

During development and testing, we identified other potential sensing opportunities that we leave to future work. For example, there are other visual appearance aspects that could be captured, including beards and hairstyle. Shoes often lie at the extreme periphery of our spherical view, and thus have limited resolution. Nonetheless, it may be possible to extract color and category for further avatar personalization. As mentioned previously, it is fairly straightforward to extract a patch of clothing to capture the pattern, though we found it challenging to tile elegantly. Finally, we also noticed that our wide-angle view permitted keypoint tracking of other people, including to the sides and even partially behind the wearer. This could have uses in e.g., social VR and spatial audio, and might even be used to provide body/hands/face data to other users without MeCap capabilities.

CONCLUSION

We have described MeCap, which enables a commodity VR headset to be augmented with powerful MoCap and user-sensing capabilities at very low cost. Using only a pair of worn hemispherical reflectors and the existing rear camera of a smartphone, MeCap provides details about the user's facial expression, physical appearance, body pose, hand gestures, and surrounding environment – capabilities which are either absent in contemporary AR/VR systems or which require specialized hardware and controllers.

ACKNOWLEDGMENTS

We are grateful to the Packard and Sloan Foundations for supporting research in the Future Interfaces Group.

REFERENCES

- [1] Karan Ahuja, Rahul Islam, Varun Parashar, Kuntal Dey, Chris Harrison, and Mayank Goel. 2018. Eye-SpyVR: Interactive Eye Sensing Using Off-the-Shelf, Smartphone-Based VR Headsets. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 2, Article 57 (July 2018), 10 pages. DOI: <https://doi.org/10.1145/3214260>
- [2] AltspaceVR, Retrieved: 12-Sep-2018, <https://altvr.com>
- [3] Brian Amento, Will Hill, and Loren Terveen. 2002. The sound of one hand: a wrist-mounted bio-acoustic fingertip gesture interface. In *CHI '02 Extended Abstracts on Human Factors in Computing Systems (CHI EA '02)*. ACM, New York, NY, USA, 724-725. DOI: <https://doi.org/10.1145/506443.506566>
- [4] Thomas Augsten, Konstantin Kaefer, René Meusel, Caroline Fetzer, Dorian Kanitz, Thomas Stoff, Torsten Becker, Christian Holz, and Patrick Baudisch. 2010. Multitoe: high-precision interaction with back-projected floors based on high-resolution multi-touch input. In *Proceedings of the 23rd annual ACM symposium on User interface software and technology (UIST '10)*. ACM, New York, NY, USA, 209-218. DOI: <https://doi.org/10.1145/1866029.1866064>
- [5] Alan Bränzel, Christian Holz, Daniel Hoffmann, Dominik Schmidt, Marius Knaust, Patrick Lühne, René Meusel, Stephan Richter, and Patrick Baudisch. 2013. GravitySpace: tracking users and their poses in a smart room using a pressure-sensing floor. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 725-734. DOI: <https://doi.org/10.1145/2470654.2470757>
- [6] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7291-7299. 2017. DOI: <https://doi.org/10.1109/CVPR.2017.143>
- [7] Chollet, François. “Keras” (2015).
- [8] Dance Central, Xbox 360. (2012). Microsoft Studios, Redmond, Washington.
- [9] Paul Debevec. Image-based lighting. *IEEE Computer Graphics and Applications* 22, no. 2 (2002): 26-34. DOI: <https://doi.org/10.1109/38.988744>
- [10] Paul Debevec. 2008. Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *ACM SIGGRAPH 2008 classes (SIGGRAPH '08)*. ACM, New York, NY, USA, Article 32, 10 pages. DOI: <https://doi.org/10.1145/1401132.1401175>
- [11] Dissanayake, MWM Gamini, Paul Newman, Steve Clark, Hugh F. Durrant-Whyte, and Michael Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on robotics and automation* 17, no. 3 (2001): 229-241. DOI: <https://doi.org/10.1109/70.938381>
- [12] 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 (UIST '16)*. ACM, New York, NY, USA, 861-872. DOI: <https://doi.org/10.1145/2984511.2984563>
- [13] 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 (UIST '17)*. ACM, New York, NY, USA, 553-563. DOI: <https://doi.org/10.1145/3126594.3126615>
- [14] Google Cardboard. (2014). <https://vr.google.com/cardboard>
- [15] Google Daydream. (2016). <https://vr.google.com/daydream>
- [16] Sehoon Ha, Yunfei Bai, and C. Karen Liu. 2011. Human motion reconstruction from force sensors. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '11)*, ACM, New York, NY, USA, 129-138. DOI: <https://doi.org/10.1145/2019406.2019424>
- [17] Chris Harrison, Desney Tan, and Dan Morris. 2010. Skininput: appropriating the body as an input surface. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 453-462. DOI: <https://doi.org/10.1145/1753326.1753394>
- [18] 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)*. ACM, New York, NY, USA, 441-450. DOI: <https://doi.org/10.1145/2047196.2047255>
- [19] Wolfgang Heidrich. "Environment Maps and Their Applications." Max-Planck-Institute for Computer Science, Saarbrücken, Germany 19 (2000).
- [20] HoloKit. <https://holokit.io>
- [21] HTC Vive. <https://www.vive.com>
- [22] PoseNet on TensorFlow Lite. <https://github.com/tensorflow/tfjs-models/tree/master/posenet>
- [23] 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)*. ACM, New York, NY, USA, 167-176. DOI: <https://doi.org/10.1145/2380116.2380139>
- [24] Chloe LeGendre, Xueming Yu, Dai Liu, Jay Busch, Andrew Jones, Sumanta Pattanaik, and Paul Debevec. 2016. Practical multispectral lighting reproduction. *ACM Trans. Graph.* 35, 4, Article 32 (July 2016), 11 pages. DOI: <https://doi.org/10.1145/2897824.2925934>
- [25] Cheng Li, and Kris M. Kitani. Pixel-level hand detection in ego-centric videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3570-3577. 2013. DOI: <https://doi.org/10.1109/CVPR.2013.458>
- [26] Hao Li, Laura Trutoiu, Kyle Olszewski, Lingyu Wei, Tristan Trutna, Pei-Lun Hsieh, Aaron Nicholls, and Chongyang Ma. 2015. Facial performance sensing head-mounted display. *ACM Trans. Graph.* 34, 4, Article 47 (July 2015), 9 pages. DOI: <https://doi.org/10.1145/2766939>
- [27] Yingliang Ma, Helena M. Paterson, and Frank E. Pollick. A motion capture library for the study of identity, gender, and emotion perception from biological motion. *Behavior research methods* 38.1 (2006): 134-141. DOI: <https://doi.org/10.3758/BF03192758>
- [28] Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. 2017. VNect: real-time 3D human pose estimation with a single RGB camera. *ACM Trans. Graph.* 36, 4, Article 44 (July 2017), 14 pages. DOI: <https://doi.org/10.1145/3072959.3073596>
- [29] David Merrill and Hayes Raffle. 2007. The sound of touch. In *ACM SIGGRAPH 2007 posters (SIGGRAPH '07)*. ACM, New York, NY, USA, Article 138. DOI: <https://doi.org/10.1145/1280720.1280871>
- [30] METAmotion, <http://metamotion.com/gypsy/gypsy-motion-capture-system.htm>, Retrieved: 12-Sep-2018
- [31] Branislav Micušik and Tomáš Pajdla. Autocalibration & 3D reconstruction with non-central catadioptric cameras. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 58-65. 2004. DOI: <https://doi.org/10.1109/CVPR.2004.1315014>
- [32] Shree K. Nayar. Sphero: Determining depth using two specular spheres and a single camera. *Optics, Illumination, and Image Sensing for Machine Vision III*. Vol. 1005. International Society for Optics and Photonics, 1989. DOI: <https://doi.org/10.1117/12.949051>
- [33] Stepan Obdrzalek, Gregorij Kurillo, Ferda Ofli, Ruzena Bajcsy, Edmund Seto, Holly Jimison, and Michael Pavel. Accuracy and robustness of Kinect pose estimation in the context of coaching of elderly population. In *Engineering in medicine and biology society (EMBC), 2012 annual international conference of the IEEE*, vol. 28, pp. 1188-1193. IEEE, 2012. DOI: <https://doi.org/10.1109/EMBC.2012.6346149>
- [34] Oculus. <https://www.oculus.com>
- [35] Kyle Olszewski, Joseph J. Lim, Shunsuke Saito, and Hao Li. 2016. High-fidelity facial and speech animation for VR HMDs. *ACM Trans. Graph.* 35, 6, Article 221 (November 2016), 14 pages. DOI: <https://doi.org/10.1145/2980179.2980252>
- [36] OptiTrack, <http://optitrack.com/>(Retrieved:12-Sep-18)
- [37] George Papandreou, Tyler Zhu, Liang-Chieh Chen, Spyros Gidaris, Jonathan Tompson, and Kevin Murphy. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric

- embedding model. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 269-286. 2018. DOI: https://doi.org/10.1007/978-3-030-01264-9_17
- [38] Helge Rhodin, Christian Richardt, Dan Casas, Eldar Insafutdinov, Mohammad Shafiei, Hans-Peter Seidel, Bernt Schiele, and Christian Theobalt. 2016. EgoCap: egocentric marker-less motion capture with two fisheye cameras. *ACM Trans. Graph.* 35, 6, Article 162 (November 2016), 11 pages. DOI: <https://doi.org/10.1145/2980179.2980235>
- [39] Daniel Roetenberg, Henk Luinge, and Per Slycke. "Xsens MVN: full 6DOF human motion tracking using miniature inertial sensors." Xsens Motion Technologies BV, Tech. Rep 1 (2009).
- [40] Silonie Sachdeva. Fitzpatrick skin typing: applications in dermatology. *Indian Journal of Dermatology, Venereology, and Leprology* 75, no. 1 (2009): 93. DOI: <https://doi.org/10.4103/0378-6323.45238>
- [41] Samsung GearVR (2015). <https://www.samsung.com/global/galaxy/gear-vr/>
- [42] 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)*, Hugues Hoppe (Ed.) ACM, New York, NY, USA, Article 31, 10 pages. DOI: <https://doi.org/10.1145/1964921.1964926>
- [43] Karen Simonyan, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).
- [44] Sony Playstation VR (2016). <https://www.playstation.com/explore/playstation-vr/>
- [45] Jochen Tautges, Arno Zinke, Björn Krüger, Jan Baumann, Andreas Weber, Thomas Helten, Meinard Müller, Hans-Peter Seidel, and Bernd Eberhardt. 2011. Motion reconstruction using sparse accelerometer data. *ACM Trans. Graph.* 30, 3, Article 18 (May 2011), 12 pages. DOI: <https://doi.org/10.1145/1966394.1966397>
- [46] TechCrunch. Retrieved: 12-Sep-2018, <https://techcrunch.com/2016/10/06/facebook-social-vr/>
- [47] Daniel Thalmann, Hui Liang, and Junsong Yuan. "First-Person Palm Pose Tracking and Gesture Recognition in Augmented Reality." In *International Joint Conference on Computer Vision, Imaging and Computer Graphics*, pp. 3-15. Springer, Cham, 2015. DOI: https://doi.org/10.1007/978-3-319-29971-6_1
- [48] Justus Thies, Michael Zollhöfer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. 2018. FaceVR: Real-Time Gaze-Aware Facial Reenactment in Virtual Reality. *ACM Trans. Graph.* 37, 2, Article 25 (June 2018), 15 pages. DOI: <https://doi.org/10.1145/3182644>
- [49] Vicon, <https://vicon.com>, Retrieved: 12-Sep-2018
- [50] VR Box. 2018. VR Box Headset. (2018). <https://www.amazon.com/Virtual-Reality-Headset-Glasses-Smartphones/dp/B01IXLEJCM>
- [51] VRChat. Retrieved: 12-Sep-2018, <https://vrchat.net>
- [52] Yan Yukang, Chun Yu, Xiaojuan Ma, Xin Yi, Ke Sun, and Yuanchun Shi. 2018. VirtualGrasp: Leveraging Experience of Interacting with Physical Objects to Facilitate Digital Object Retrieval. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Paper 78, 13 pages. DOI: <https://doi.org/10.1145/3173574.3173652>
- [53] Mingmin Zhao, Tianhong Li, Mohammad Abu Alsheikh, Yonglong Tian, Hang Zhao, Antonio Torralba, and Dina Katabi. Through-wall human pose estimation using radio signals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7356-7365. 2018. DOI: <https://doi.org/10.1109/CVPR.2018.00768>
- [54] Cheng Zhang, Qiuyue Xue, Anandghan Waghmare, Ruichen Meng, Sumeet Jain, Yizeng Han, Xinyu Li, Kenneth Cunefare, Thomas Ploetz, Thad Starner, Omer Inan, and Gregory D. Abowd. 2018. FingerPing: Recognizing Fine-grained Hand Poses using Active Acoustic On-body Sensing. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Paper 437, 10 pages. DOI: <https://doi.org/10.1145/3173574.3174011>
- [55] 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 (UIST '15)*. ACM, New York, NY, USA, 167-173. DOI: <https://doi.org/10.1145/2807442.2807480>
- [56] Yang Zhang, Chouchang (Jack) Yang, Scott E. Hudson, Chris Harrison, and Alanson Sample. 2018. Wall++: Room-Scale Interactive and Context-Aware Sensing. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Paper 273, 15 pages. DOI: <https://doi.org/10.1145/3173574.3173847>
- [57] Junhan Zhou, Yang Zhang, Gierad Laput, and Chris Harrison. 2016. AuraSense: Enabling Expressive Around-Smartwatch Interactions with Electric Field Sensing. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 81-86. DOI: <https://doi.org/10.1145/2984511.2984568>