PatternTrack: Multi-Device Tracking Using Infrared, Structured-Light Projections from Built-in LiDAR

Daehwa Kim
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
daehwak@cs.cmu.edu

Robert Xiao
Department of Computer Science
University of British Columbia
Vancouver, British Columbia, Canada
brx@cs.ubc.ca

Chris Harrison
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania, USA
chris.harrison@cs.cmu.edu

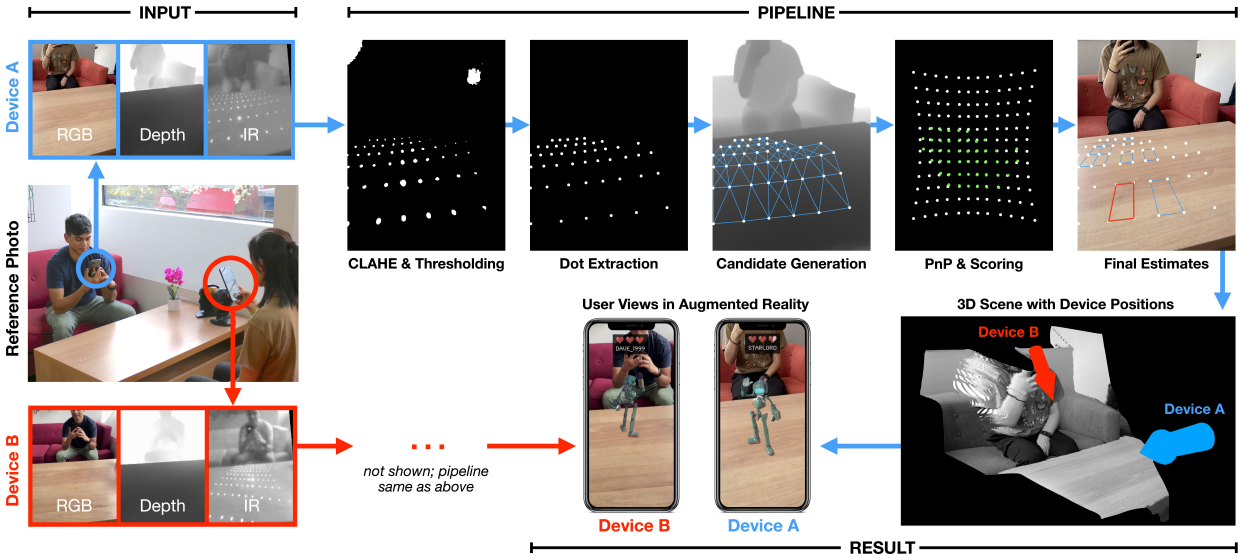


Figure 1: Consumer devices with LiDAR depth sensors are becoming more common (e.g., iPhone, Vision Pro). These sensors emit an infrared structured light pattern as part of their operation. The core insight of PatternTrack is to capture the projected patterns of other proximate devices; the geometric information inherent in the patterns allows PatternTrack to estimate the 6DOF position of other co-located devices. In this example scene, two users are playing an AR game on an ordinary coffee table.

Abstract

As augmented reality devices (e.g., smartphones and headsets) proliferate in the market, multi-user AR scenarios are set to become more common. Co-located users will want to share coherent and synchronized AR experiences, but this is surprisingly cumbersome with current methods. In response, we developed PatternTrack, a novel tracking approach that repurposes the structured infrared light patterns emitted by VCSEL-driven depth sensors, like those found in the Apple Vision Pro, iPhone, iPad, and Meta Quest 3. Our approach is infrastructure-free, requires no pre-registration, works on featureless surfaces, and provides the real-time 3D position and orientation of other users' devices. In our evaluation — tested on six different surfaces and with inter-device distances of up to 260 cm — we found a mean 3D positional tracking error of 11.02 cm and a mean angular error of 6.81°.

CHI '25, Yokohama, Japan

ACM ISBN 979-8-4007-1394-1/25/04
https://doi.org/10.1145/3706598.3713388

Keywords

Multi-device tracking, augmented reality, co-located collaboration.

ACM Reference Format:
Daehwa Kim, Robert Xiao, and Chris Harrison. 2025. PatternTrack: Multi-Device Tracking Using Infrared, Structured-Light Projections from Built-in LiDAR. In *CHI Conference on Human Factors in Computing Systems (CHI '25), April 26-May 1, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3706598.3713388

1 Introduction

As augmented reality (AR) experiences go mainstream, the likelihood of co-located users wishing to participate in shared, coherent experiences is growing. Unfortunately, initiating a shared AR experience is surprisingly cumbersome using current methods. As a result, few apps incorporate such functionality, and most of these are small "toy" experiences. This stands in stark contrast to the decades-old envisionment of groups of doctors, engineers, and other professionals wearing headsets and interacting around shared, complex tasks (see examples in Figure 2). Thus we believe the need for new approaches in multi-device localization has never been greater.

We found that the three most common approaches suffer from at least one significant downside. The first option is using fiducial

Figure 2: Multi-user synchronized AR experiences have been shown in innumerable marketing materials — like these shown here — and yet few users have ever experienced such applications. We believe the rarity of these experiences is due to the cumbersome nature of existing multi-user pairing/tracking methods, and thus there is a great need to identify new approaches.

markers [19, 27, 44, 81, 85]. While reliable and effective, they require a physical printout, which obviously requires preparation and distracts from the scene. Second, we have UWB- [9, 74] and Bluetooth-derived proximity [39], as well as RF localization technologies [26, 42, 45, 56, 74]. However, these techniques generally require external infrastructure, or only provide distance estimates (but not vector or positional information we need). Finally, and perhaps most promising from a user experience standpoint, is for each device to perform a brief scan of the environment and then exchange spatial data to correlate their relative positions [7, 8, 51]. However, this brief scan is not instant — for example, Apple's ARKit takes several seconds to create a paired experience each time. If the desired interaction surface is featureless, as many walls are, it either fails during setup or requires more spatial data. Additionally, devices need to exchange data, which users may not wish to do in transient contexts.

In response, we developed PatternTrack (Figure 1), a new multi-device localization approach with a mix of attributes that differentiate it from prior methods (Figure 3). In short, PatternTrack:

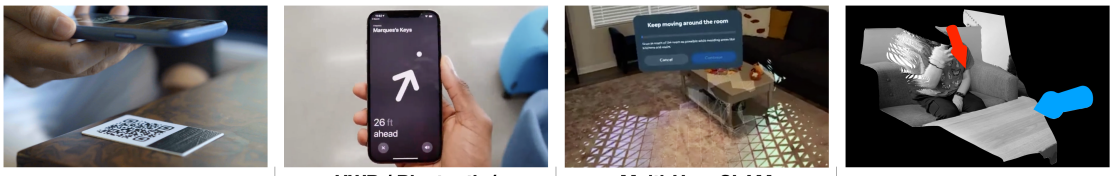
(1) Requires no external props of infrastructure to operate (no printed markers, no base stations, etc.) other than commonplace passive surfaces (e.g., walls and tables).
(2) Works on featureless surfaces (e.g., painted wall, clean whiteboard).
(3) Can establish a shared session with as little as a single frame of data, with no preregistration process.
(4) Works at typical co-located collaboration ranges (0.5-2.5 m).
(5) Provides real-time 3D position and orientation of other users' devices, and is spatially accurate down to about the size of a smartphone.

The core insight of our approach is to repurpose the structured light patterns emitted by depth sensors found in devices such as the Apple Vision Pro, iPhone, iPad, and Meta Quest 3. In essence, these devices contain a miniature infrared projector that emits a known pattern. Other co-located devices can see this pattern and its characteristic perspective distortion on surfaces that fundamentally reveal the 3D vector and distance of the projecting device. Figures 1 and 4, and our Video Figure, provide illustrative examples of this phenomenon.

After briefly reviewing related work, we describe our software implementation. Because of third-party API limitations, we had to build custom hardware to simulate what is available to OEMs such as Apple and Meta. We then describe our evaluation and its results, which show that our approach offers around 11.02 cm positional accuracy and 6.81° angular error when utilizing just a single frame of data.

2 Related Work

In the previous section, we briefly discussed three common approaches used in commercial software to support multi-user co-located XR. Even still, few apps incorporate such functionality — at the time of writing, there are less than a dozen apps on the Meta Store for Oculus headsets, and no such apps on Apple's App



	Fiducial & Visual Markers	UWB / Bluetooth / Ultrasonic / etc. Tracking	Multi-User SLAM (w/ shared geometry)	PatternTrack (ours)
Setup	Requires physical prop 👎	Requires basestations for 6DOF tracking; ≤3DOF without 👎	Users must scan area first 👎	No Setup 👍
Works on Featureless Surfaces	No (requires marker) 👎	Yes 👍	No (users must scan beyond featureless area) 👎	Yes 👍
Initialization Time	Instant (single frame) 👍	Near Instant (few seconds) 👍	>5 seconds 👎	Instant (single frame) 👍
Tracking Distance	Short to Medium (depends on marker size) 👍	Far (building-scale) 👍	Far (building-scale) 👍	Medium (0.5-2.5 m) 👍
Tracking Resolution	Very Accurate (~1 cm) 👍	Coarse (~1 m) 👎	Accurate (~4 cm) 👍	Accurate (~11 cm) 👍

Figure 3: High-level overview of contemporary methods for initiating multi-user AR experiences.

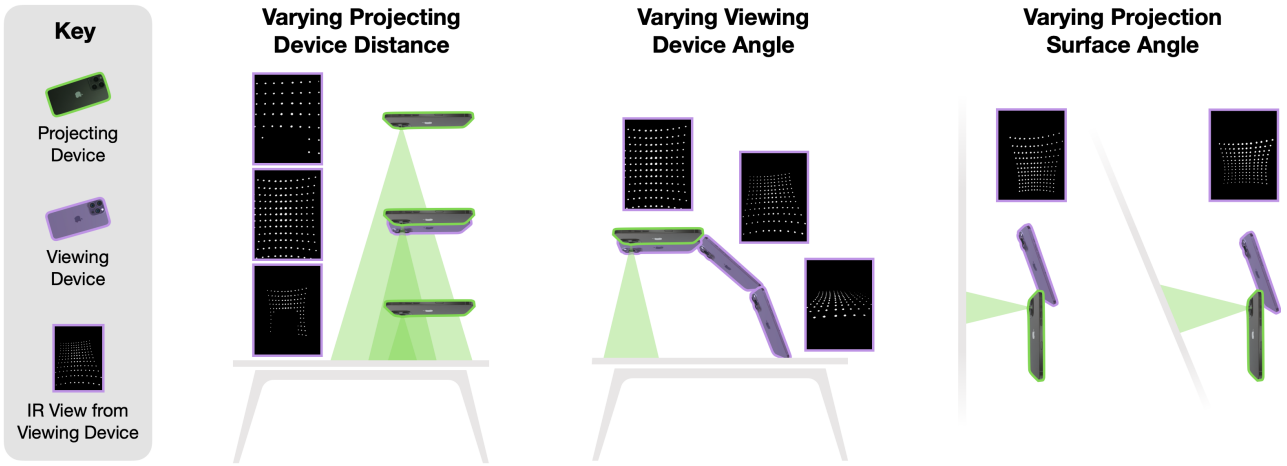


Figure 4: The appearance of the projected pattern depends on the 6DOF position/rotation of the viewing device, projecting device, and surface. In this figure, we show how three exemplary degrees of freedom impact the viewed pattern.

Store for its Vision Pro headset. In this section, we instead focus on research approaches and emerging techniques.

2.1 Multi-User Interaction & AR

Applications in augmented reality span domains as diverse as education [21, 60], entertainment [4, 54, 89], office [11], travel [33], urban planning [85], industrial [86], and social media [5, 23] (see also Figure 2). Collaborative AR allows these experiences to be shared among users in co-located settings, and has several unique characteristics when compared with single-user AR.

The computer-supported collaborative work (CSCW) community has long identified interaction requirements based on the nature of cooperation [71]. In a collaborative context, the interaction is not just about viewing common objects, but also involves sharing critical information for communication [24, 30]. This can include tracking another user's viewpoint [82], maintaining joint attention [14], and accessing user-centric UI elements (e.g., health points in games). Instantaneous and continuous device/user localization, ideally without pre-registration, is critical to the user experience [55]. There is also considerable work in the area of cross-device interaction; Brudy et al. [16] presented an excellent summary of cross-device tracking characteristics and modalities and we refer readers to this paper for additional overview (see Table 2 in [16]).

Also relevant to our work are sociological theories, including proxemics [18, 32], F-formations [20, 40], and micro-mobility [46]. Hall [32] famously proposed social zones in research on proxemics: *Intimate* (0 ~ 0.2 m), *Personal* (0.3 ~ 1.2 m), *Social* (1.2 ~ 3.7 m), and *Public* (3.7 ~ 7.6 m) distances. Although impersonal business collaboration usually happens within the *Social* range, the *Personal* bubble is more common for collaboration with colleagues. The smaller personal space is also used for social interactions among friends [18, 32]. When in an F-formation, typical inter-participant distances are around 1 meter, even for groups of up to seven people [4, 34] (see also [49, 70] for a discussion on other small-scale, most typically <1 meter, social configurations, such as O-Spaces, p-Spaces, N-shapes, etc.). Figure 2 shows several example scenes with small groups of users in such formations. More specific to the HCI domain, Marquardt et al. [49] explored user configurations during collaboration with handheld devices across different tasks, including face-to-face (competitive), side-by-side (collaborative), and corner-to-corner (communicative).

2.2 Co-Located, Infrastructure-Required, Multi-Device Tracking

While there exists a large body of methods for tracking a single mobile device in space [2, 28, 41, 64], in this section we specifically focus on technologies and challenges for tracking multiple co-located devices.

Popular systems such as OptiTrack [58] and Vicon [83] achieve accurate tracking by placing fixed cameras in the environment looking for optical markers. AR applications have been enabled using this approach [13, 48], though it is expensive and immobile. Without using optical markers, fixed depth cameras in a room have been shown to achieve 3D tracking with around 7 cm spatial error [91]. It is also possible to place base stations in an environment that emits a known signal, most often light. For example, the VIVE Lighthouse system [84] used timed infrared laser swipes and triangulation to localize one or more active trackers in a scene. Instead of using time, Lumitrack [92] used projected light patterns to allow mobile optical sensors to localize themselves in a scene. CLIPS [80] installed a stationary "laser hedgehog" that projected a structured light pattern onto the ceiling, with which devices could localize themselves using a camera. In some respects, the latter two approaches are the most technically similar to PatternTrack, in that both systems view a light pattern for tracking. However, PatternTrack uniquely repurposes an existing pattern emitted for a totally different purpose (LiDAR depth sensing) and does not require special receivers or emitters. We also note that PatternTrack works with partial patterns, which is a frequent occurrence in real-world scenes (occlusion from objects, falling off edges of furniture, projecting onto dark materials that absorb infrared light, etc.).

Fiducial markers are another popular approach to enable six degrees-of-freedom (6DOF) tracking. Devices instrumented with these markers can be tracked by other devices [1], or many devices can track one or more markers placed in the environment (with each device responsible for determining its own relative position to markers, which can then be shared with all participating devices) [85]. This is similar to projector-camera calibration approaches, which often utilize a physical calibration board with a printed pattern. Chan et al. [19] made these fiducial markers invisible to the human eye by projecting them in infrared. Non-optical methods are also possible, such as electromagnetic tracking [53, 61, 95], which can be very accurate, but operate in smaller volumes. For larger scales, RF methods (e.g., RFID/NFC/Bluetooth/WiFi; fingerprinting or triangulation) [26, 42, 45, 56, 74] have been demonstrated, but tend to be less spatially accurate.

2.3 Co-Located, Infrastructure-Free, Multi-User Tracking with Off-the-Shelf Devices

A significant body of work has explored methods that enable individual devices to track their position in a 3D environment (see e.g., [50] for a survey). More relevant to this work are techniques that utilize projection in some manner, including laser speckle optical flow [98] and various structured-light-based depth estimation techniques [31, 43, 62, 63]. With such tracking information, many devices could theoretically share their positional data among themselves to create a multi-user experience. However, very few research papers extend their work to this final step.

The most widely used method at present is multi-user Simultaneous Localization and Mapping (SLAM), which involves exchanging scanned scenes between two or more devices [7, 8, 51]. These scenes are interrogated to find matching geometry to create a shared spatial understanding, with each device placed into the 3D scene. To capture enough scene context, current software generally requires a seconds-to-minutes registration process for all participating users and is hardly seamless. Also, such methods generally demand a scene abundant in features and further necessitate that the user move the device adequately to capture enough of a scene for matching to occur between devices. Sharing scene maps between devices also presents challenges, as it depends on the availability of a shared wireless network and trusted endpoints (e.g., users may not wish to freely transmit their point cloud data to proximate strangers). Researchers [65, 97] have also worked on reducing latency while sharing and matching scene maps during the initialization of a multi-user AR session. Deep learning has also been used to estimate camera positions when presented with two or more source images [72, 87]. However, these models do not currently operate in real-time. SynchronizAR [36] combined SLAM and UWB data for multi-device tracking, but still requires pre-registration of the environment before a session can be established. Using the SLAM moniker in a different way, BodySLAM [1] employed user-worn fiducial markers to track multiple users, but which must appear in the camera views of at least one other user. We note that PatternTrack is effective even when devices are not directly facing each other, a common scenario in AR where users are looking at shared virtual content and not one another.

Moving to nonvisual methods, Ultra Wide Band (UWB; available in some newer smartphones) [9] and Bluetooth [35, 39] have been used to estimate a device's rough distance relative to another participating device. These methods do not provide 6DOF tracking that is needed for finer-grained AR interactions (in the unique case of Apple's UWB implementation, it is combined with visual odometry to create a more precise AR-like experience, but which requires many UWB readings integrated over time to improve spatial accuracy). Tracko [38] utilizes both BLE and acoustic tracking; inaudible sound roundtrips between devices allow Tracko to calculate distances between speaker-microphone pairs, but does not estimate orientations.

More similar application-wise to PatternTrack is SideBySide [90], which demonstrated a prototype handheld consumer device that used an infrared fiducial marker projection to enable multi-user interactions on ad hoc surfaces such as walls. We note, however, that this work never attempted to estimate the 6DOF position of other projecting devices (instead resolving only relative 2D position between devices on a projection surface). SideBySide also required modifications to an off-the-shelf infrared projector, whereas PatternTrack uses the existing infrared pattern already emitted by some consumer devices. The application domains are also different, with PatternTrack focusing more on augmented reality and 3D interactions between devices. Nonetheless, the end-user simplicity and seamlessness of the method is something we sought to replicate in PatternTrack.

In summary, different methods applicable for multi-user AR have varying pros and cons. We provide an overview of the most widely used methods in Figure 3. PatternTrack offers a unique blend of advantageous properties: it is infrastructure-free, provides accurate 6DOF tracking, operates on both featured and featureless surfaces, does not require a calibration/registration step, and requires only a single frame to estimate device positions. However, it is not as far-ranging as some other methods, and for this reason, we envision it for use in typical collaboration distances. In our evaluation, we test up to ranges of 1.5 m from surfaces, and 2.6 m inter-device (i.e., inter-user) distances. This represents the typical envelope of distances that users might stand from a table or wall when collaborating in AR (see Section 2.1 for this discussion).

2.4 Multi-Projector & Projector-Camera Calibration

We note that multi-projector display work (for e.g., projected augmented reality, environment projection mapping) requires related spatial registration methods, and has spawned many methods [15, 66–68, 79]. Likewise, there is a large body of work that requires projector-camera calibration, which faces similar registration issues (in essence, a camera is a reverse projector). For calibration, a physical object with a printed pattern or 3D geometry is often used. Alternatively, patterns can be emitted by projectors [66], for calibration on both planner [68] and non-planner [15] surfaces. iLamps [67] integrated a projector and a camera into a single self-contained unit and presented a shape-adaptive projection technique to improve registration performance. It is also possible to use many cameras and many projectors for registration, as demonstrated in Tehrani et al. [79]. Today, these methods are mature; PatternTrack

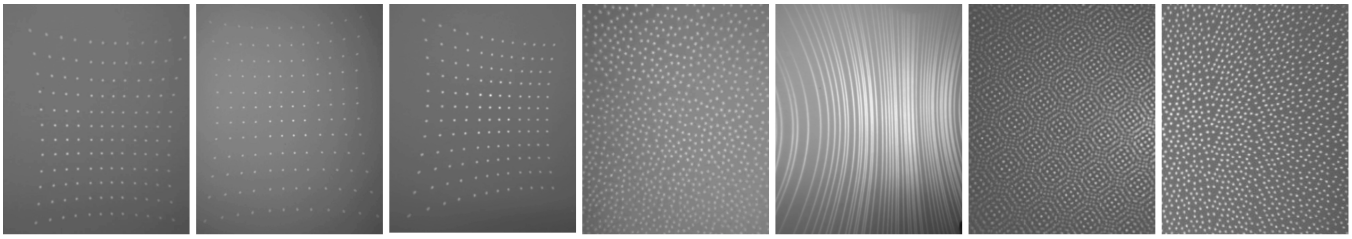


Figure 5: Example infrared patterns emitted by various consumer devices using vertical-cavity surface-emitting lasers (VCSELs).

builds on many of the fundamental computer vision and graphics techniques from this literature. Uniquely, PatternTrack co-opts the existing patterns already projected by several popular consumer devices (e.g., iPhones), and works with mobile/worn devices that are in constant motion (whereas much of the prior work was fixed and calibrated once).

3 PatternTrack

We now describe our hardware and software pipeline. We note that to fully realize PatternTrack, one would need low-level access to hardware/firmware, which is typically only available to device OEMs. As such, our implementation serves as a proof-of-concept to demonstrate the core algorithm.

3.1 Proof-of-Concept Hardware

We used iPhones 13 Pro, 12 Pro Max, and 14 Pro Max throughout development. These devices contain Apple's proprietary, rear-facing direct time-of-flight LiDAR (Light Detection and Ranging) sensor [93, 96], which utilizes 940 nm vertical-cavity surface-emitting laser (VCSEL) emitters. Infrared images of these emitters are shown in Figure 5, along with example patterns from other depth-sensing devices. Geometrically, this pattern consists of a set of rays emanating from the projector's focal point, and can also be represented visually as a set of points on an imaginary image plane at z=1. The pattern consists of a total of 144 dots, divided into nine smaller 4×4 grids with a small gap between each grid. Our reverse-engineered model of the pattern, shown on the image plane, is shown in Figure 8 (pattern model).

Apple only provides an API for the processed depth map, but not the infrared image. Public "teardowns" have indicated that the resolution of the Sony infrared sensor used by Apple is 30K pixels, or approximately 200×150 (see e.g., [93, 96] for more information on these sensors). In lieu of software access in iOS to the infrared image, we instead affixed an off-the-shelf infrared camera (p.n. Raspberry Zero V1.0 mini Camera) to the rear of the phone (fitted with a 940 nm bandpass filter, matching the wavelength of the iPhone's LiDAR sensor). We use a RaspberryPi Zero 2 W to capture and stream 640×360 video to a host computer over WiFi, where it is merged with the iPhone's rear-facing wide-angle RGB camera [6] and depth map data, also streamed over WiFi to the laptop. iOS provides functionality to align the depth map to the RGB camera [10], and thus we also align our infrared camera to the RGB stream using a homography computed at a range of 60 cm, providing aligned RGB+D+IR frames. However, as the device gets closer or further away from this calibrated distance, it introduces a small systematic bias in our position estimates (due to parallax), and so we apply a linear correction to our position predictions to account for this. Our prototype hardware is seen in Figures 1 and 6.

3.2 Operational Range & User Formations

As discussed in Related Work, research over many decades into proxemics [32, 34, 49, 70, 75] has identified that colleagues typically collaborate within a 1.2 m range. For example, two collaborators standing in front of a whiteboard or sitting across from one another playing a board game (other examples shown in Figure 2). For this reason, our prototype implementation and later evaluations targeted this especially high value range. Longer ranges are possible with PatternTrack but would require a higher resolution camera to detect patterns at longer ranges (our current infrared camera streams at just 640×360). Also, as discussed in Related Work, users naturally form into specific social formations around tasks that tend to avoid occlusion for participants. For instance, two users might be side-by-side when facing a shared surface, such as a wall or whiteboard, or in a circular arrangement around a table. In general, PatternTrack is amenable to these social configurations, as users already desire unoccluded line of sight and even physical reachability to shared work surfaces (both vertical and horizontal).

3.3 Applicable Devices & Cross-Device Uses

We note that LiDAR and VCSEL infrared pattern emitters are becoming more common in consumer electronic devices, including smartphones, tablets, and XR headsets, such as iPhone Pro models, iPad Pro, Samsung's Galaxy S20, Huawei P30 Pro, Apple Vision Pro and Meta Quest 3. Although we can see the LiDAR patterns of these devices (Figure 5), we leave the implementation of PatternTrack for these platforms to future work. Nonetheless, we are confident that PatternTrack could generalize across these platforms by detecting and loading the appropriate pattern models. Certainly all of these devices have the same general capabilities as that of the iPhones we used for our proof-of-concept implementation (RGB video, depth stream, IR emitter, reasonably robust compute, etc.).

Assuming that each participating device's pattern model could be determined (or revealed with e.g., a Bluetooth/UWB handshake), then it is imminently possible to enable cross-device-model interactions, where some users could be e.g., using a tablet, while others are wearing XR headsets of different makes. Such multi-device

Figure 6: Although the iPhone already contains an infrared camera, it is not accessible to 3rd party developers. Thus, we instrumented our iPhones with infrared cameras (plus a Raspberry PI to read the camera and stream video over WiFi).

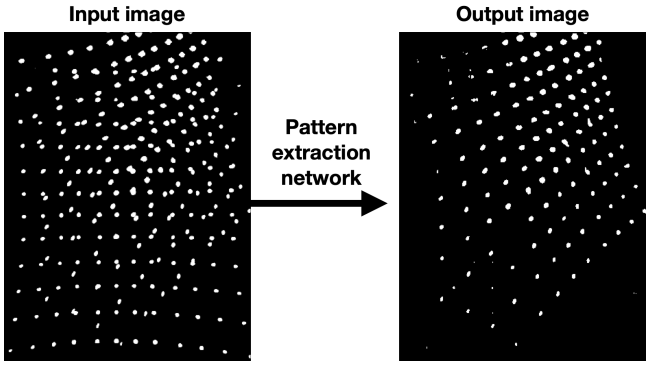


Figure 7: Left: a device's own dot pattern overlapped with the dot pattern of a co-located device. Right: As one potential method to solve the issue of overlapping patterns, we created a deep learning model that removes a device's own pattern from its infrared camera stream (example output shown here). Both images are real data.

interaction has already been motivated and explored in the HCI community (see e.g., seminal work by Brudy et al. [16] and Marquardt et al. [49]), but using other tracking technology means. In the future, PatternTrack-like methods could serve to further reduce the burden of instantiating such interactions, potentially one day making it seamless for users.

3.4 Multi-Device Pattern Extraction

In order to estimate another device's position, we must first segment its dot projection. Of course, if there are many simultaneous devices, their dot patterns will overlap on a shared surface. Thus, the first step of our pipeline is to isolate each device's pattern. Channel contention in a shared medium is a well-studied problem and numerous potential solutions exist. We implemented two different approaches.

First, because the viewing device knows its own pattern (which varies little over space because the emitter and camera move together), we trained a deep learning model to recognize a device's own pattern and subtract it from the live infrared stream (example in Figure 7). Our model architecture resembles a denoising autoencoder; the encoder has 3 convolutional layers (64 channels each, 3×3 kernels) with max pooling, followed by a decoder with 3 upsampling layers and matching convolutions. The network is trained on synthesized data: the input is an overlapped, thresholded image of two projected patterns (own and another device), while the output is an image of only the other device's projected pattern. This approach was reasonably effective at separating two interfering patterns, but does not easily scale beyond two devices.

As a second implementation, we used basic time multiplexing, where each device fires its pattern at a different time. For this, we simply start and stop an ARKit session using Apple's API, which triggers and then terminates the LiDAR sensor. If too many dots are detected, we know that patterns collided, and we can utilize exponential or random backoff (or similar channel contention methods) to desynchronize two or more devices. In a proper commercial implementation, we envision co-located devices using Bluetooth or other local communication scheme to more efficiently desynchronize their patterns. We found our time multiplexing approach to be more robust and debuggable than our machine learning method, with the added benefit of being able to scale to many co-located devices, and so we selected it for our prototype implementation.

3.5 Dot Finding & Candidate Square Extraction

Having now captured another device's pattern, the observing device segments all of the dots. We first apply OpenCV's Contrast Limited Adaptive Histogram Equalization (CLAHE) [57] to enhance and normalize the image (clipLimit=5, tileGridSize=40×40). We perform some basic filtering using blob size and eccentricity to get a list of candidate projected dots. We then utilize the phone's depth map to translate these blobs into 3D points in space.

Using our list of candidate points, we form candidate squares using a greedy matching algorithm starting at the centroid of the dot pattern. Successful candidate squares follow some basic sanity checks (e.g., not allowing points inside of their bounds, not having extreme interior angles, having a min/max size), making them more likely to be the true unit size of the projected dot grid. Additionally, because we do not know the orientation of the projecting phone, each candidate square has four possible orientations. As the total number of possible candidate squares can be large, we typically set our system to suspend candidate generation after 100 squares. Note that squares projected from an oblique angle will actually appear trapezoidal on the surface, which our heuristics must account for.

3.6 Perspective-n-Point

For each candidate square in the observed points, we attempt to pair it with a corresponding square in the ideal pattern model (Figure

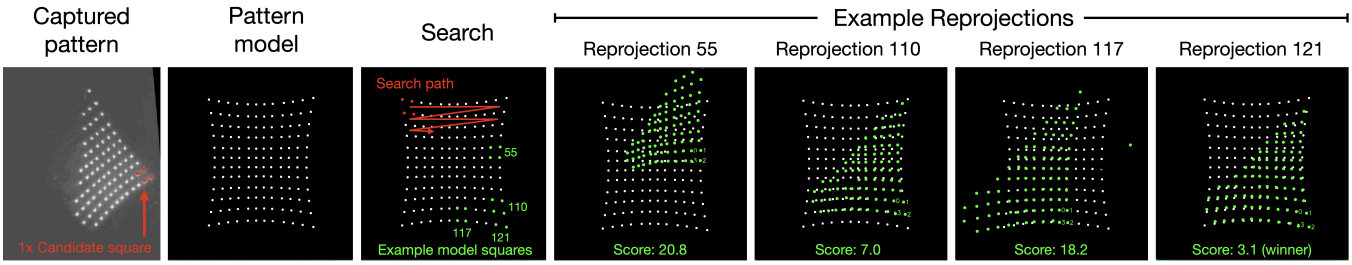


Figure 8: PatternTrack utilizes two pieces of information: a captured pattern (at 30 FPS) and the pattern model (static). For a given candidate square in the captured pattern, we test it against all squares in the pattern model (using P4P), and use the resulting transformation matrix to reproject all other detected dots (see example reprojections). We then compute the mean distance between all captured dots and their closest dot in the pattern model and use this as a match score (lower is better). PatternTrack brute forces this for all candidate squares in the captured pattern, finding the best match for the entire input frame. The winning transformation matrix reveals the projecting device's 6DOF position.

8) via a brute-force approach (testing each possible pattern square in turn). For each potential pairing between the candidate square and a pattern square, we solve the Perspective-Four-Point (P4P) problem using the Lambda Twist P3P solver [59] to obtain the pose (3DOF rotation + 3DOF position) of a projector whose four pattern rays would intersect the four points of the candidate square. With the projector pose, we then project all other observed points onto the projector's image plane, and then match each observation with its nearest pattern point. This is done efficiently by querying a precomputed KD-tree containing all of the pattern points.

Finally, the mean reprojection error (distance) between the projected observations and the pattern points, combined with the difference between the previous projector pose and the new projector pose, is used to score the potential pairing. We select the pairing with the lowest score among all pairings between candidate squares and pattern squares. We finally refine the selected projector pose by solving a Perspective-n-Point problem between the set of all observed points.

3.7 Post Processing

We apply a basic filter to our 6DOF estimates to reject outliers and increase stability. Specifically, we check if the phone has moved more than 20 cm since the last frame (approximately 33 ms, as our cameras run at 30 FPS), which would mean the phone exceeded a velocity of 6 m/s, which is unlikely. If we detect this condition, we ignore the "best" match and look at our second-best result. If this is less than 20 cm from the previous frame's estimated position, we accept it, and if not, we do not produce an estimate.

3.8 Communication Architecture

In addition to the technical pipeline illustrated in Figure 1, we also document the communication architecture of our prototype implementation in Figure 9. Of note, our prototype system employs a laptop to receive the data streams from an iPhone (depth and RGB) and its physically coupled Raspberry Pi (infrared camera), both over wifi. Of course, in a proper implementation with OEM-level access to the hardware and operating system, all functionality would be centralized into a self-contained device. The output of our process is the transformation matrix of any other participating phones.

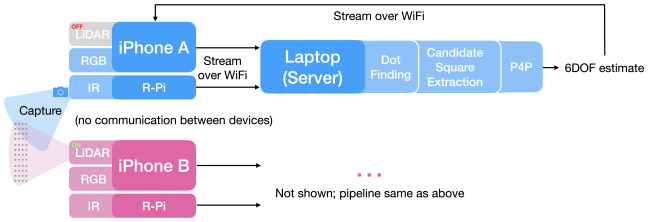


Figure 9: Communication architecture used by our proof-of-concept implementation. Note that in a commercial version, there would be no need for a laptop — the entire implementation would execute on a self-contained device such as a smartphone or headset.

3.9 Performance

The most computationally-heavy part of our pipeline (brute-force rectangle generation and P4P matching) runs at on a Macbook Pro M1 (2021) laptop at approximately 8 FPS. We report this number not as an innate limitation, but rather as a baseline performance that can be exceeded in future work. We note, for instance, that our entire pipeline is written in Python (not known for its efficiency in handling multiple video streams) and we invested only modest efforts into optimization (not the goal of this proof-of-concept work). By taking advantage of hardware-accelerated frameworks like Apple's Metal, or even just better parallelization, significant performance improvements may be possible over our current brute-force CPU-bound implementation (more discussion on this topic in our Limitations section). In general, we do not foresee any significant technical obstacles preventing PatternTrack from running in real-time at typical camera framerates (30-60Hz) with proper commercial engineering.

3.10 Example Uses

As already summarized in Related Work, the use and utility of multi-user AR has been well established in prior work. For this reason, our goal was not to create new interaction techniques for multi-user AR, but rather offer a new method to instantiate such shared AR experiences. Of course, possible use domains are numerous;

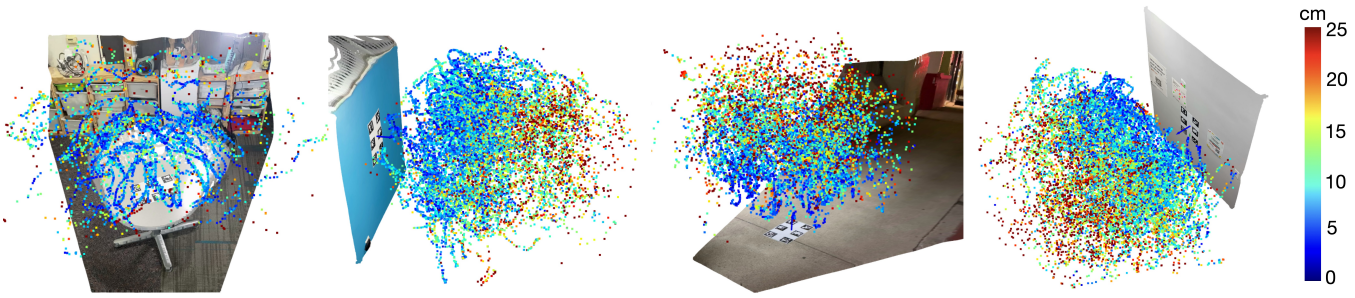


Figure 10: Our study procedure was designed to capture a variety of distances and angles between our two test iPhones, and also with respect to the projection (i.e., interaction) surface. Shown here for illustration are point clouds from four of our six surface conditions: gray table, blue wall, sidewalk, and white wall. The left-most point cloud shows only the manual serpentine path from *one* data collection session (~50 cm projecting phone distance). The other three-point clouds include data from all sessions combined (32,400 points each). Points are color-scaled by their 3D positional error. Unsurprisingly, the error increases as devices move farther away (e.g., even small errors in the P4P-predicted vector angle magnify with longer rays; see also results in Figure 14). However, unexpectedly, there was not a significant accuracy loss at oblique angles with respect to the surface or the other device (see also results in Figure 13). In total, we collected 194,400 trials for data evaluation.

we highlighted five significant examples — education, industrial, medical, entertainment, and office work — in Figure 2. Nonetheless, we did create a very small suite of demonstration interactions built on top of PatternTrack that can be seen in our Video Figure.

3.11 Open Source

To better convey smaller implementation details and facilitate replication of PatternTrack, we have open-sourced our system: *https://github.com/FIGLAB/PatternTrack*

4 Evaluation

Using our aforementioned iPhone prototypes, we designed and ran an evaluation to test key factors that influence performance, namely: the projection surface material, the distance of both the projecting and viewing phones, and the relative angle between the two devices and the surface on which the dot pattern is projected.

4.1 Setup

We used two iPhones as test devices in our evaluation. For ground truth position and orientation, we place a printed grid of ArUco markers on each test surface. Each phone can calculate its position and orientation with respect to the markers, and thus we can calculate the relative position of the two devices to evaluate the accuracy of PatternTrack's 6DOF estimations. Importantly, these printed markers are not visible in the infrared camera stream and thus did not interfere with our pipeline.

4.2 Procedure & Study Conditions

We tested PatternTrack on six commonplace surfaces: a gray office table, a wood dining room table, a vinyl tile floor, a concrete sidewalk, a blue-painted wall, and a white-painted wall with some student postings. Matched RGB+IR+Depth images of these surfaces can be seen in Figure 11. Note the dot pattern varies a little in intensity, but is well above noise. (There are some surfaces on which our technique does not work, such as glass, which we discuss later in Limitations).

To control distance and angle, the viewing iPhone was mounted on a tripod. For each surface condition, we captured data at three different distances from the surface (0.5 m, 1.0 m and 1.5 m) and at three different viewing angles (for horizontal surfaces: 30°, 60°, and 90°, the latter being perpendicular to the surface; for vertical surfaces: 0° straight on, 45° tilted down, 45° from the left). These distance and device orientation conditions were derived from prior work in multi-device collaboration (see e.g., [18, 32, 48, 70] and Section 2.1). The projecting iPhone was handheld by an experimenter and manually moved in a serpentine pattern while orbiting the surface. The experimenter also varied their surface distance from approximately 0.5 m to 1.5 m. At our shallowest angle of 30°, this means the two phones were up to 2.6 m apart when on opposite

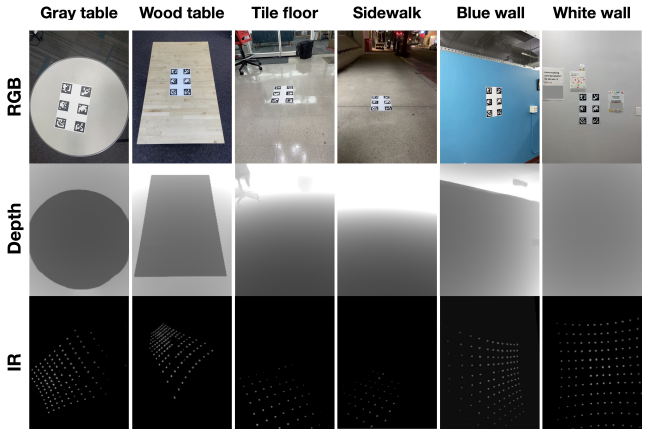


Figure 11: Example RGB/depth/infrared image sets captured during our study. Note the ArUco marker (used for 6DOF ground truth) is visible in the RGB camera, but not in the depth or infrared streams that PatternTrack uses.

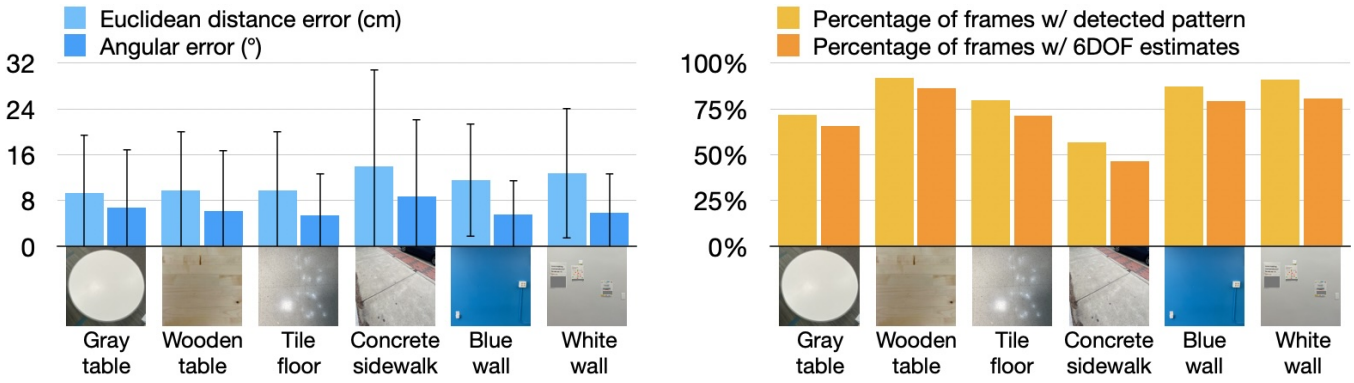


Figure 12: Study results broken out by surface condition. Left: Euclidean distance and angular error. Right: The percentage of input frames where our pipeline detected the other device's projected pattern, along with the percentage of frames that made it through our whole pipeline and produced a 6DOF estimate (i.e., not filtered along the way due to a detected error condition).

sides of a surface (well beyond a typical collaboration distance and into the social proxemics sphere). Our procedure also collected data when the two phones were side-by-side and almost touching. This procedure was purposely designed to capture data within a typical envelope of distances between two users who might stand at a table or wall when collaborating in AR (see Section 2.1 for more discussion).

The tripod-mounted iPhone was used to capture two minutes of data at 30 FPS, resulting in 3600 frames of data in each session. The above process was repeated for each surface-distance-angle condition. Thus our 6 surfaces, 3 viewing angles, and 3 viewing distances yielded 194,400 frames for analysis. Example points clouds of captured data are shown in Figure 10, which we note are both spatially varied and dense.

5 Results

We evaluated our system using four metrics; (1) Euclidean distance error between ground truth and predicted device positions, (2) angular error between ground truth and predicted orientations, (3) the percentage of captured frames in which our pipeline detected a pattern, and (4) the percentage of frames in which our pipeline produced a 6DOF estimate.

Overall, we found a mean positional tracking error of 11.02 cm (SD=11.57). For size reference, this is substantially smaller than the screens on the iPhones we used for the experiment (i.e., the predicted position is a good estimate of a phone's location). In estimating device orientation, we found a mean angular error of 6.81° (SD=9.00). In the following subsections, we discuss the effects of surface material, device distances, and device angles.

Of important note, our data streams are not synchronized. Each iPhone streams its time-aligned RGB camera and depth map to a laptop over WiFi. At the same time, each phone's Raspberry Pi Zero 2 W also streams infrared camera frames over WiFi. Due to image compression and conversion overhead, WiFi contention, and packet loss in general (causing TCP/IP to retransmit packets), we occasionally observed asynchronies between the streams in excess of one second. This latency introduces a discrepancy between the ground truth and the estimated 6DOF position/rotation. If we align

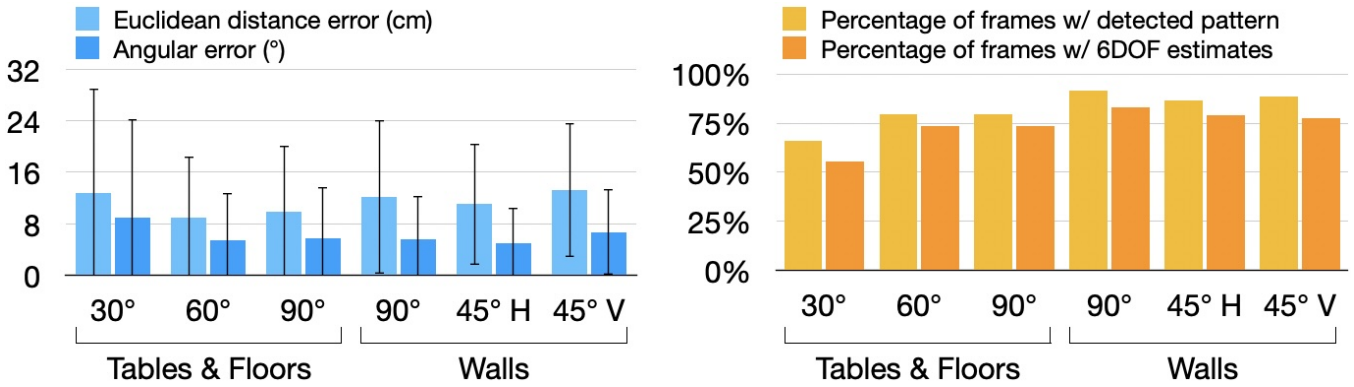


Figure 13: Study results broken out by the viewing phone's angle with respect to the surface. Left: Euclidean distance and angular error of our horizontal and vertical surface condition. Right: The percentage of input frames where our pipeline detected the other device's projected dots, along with the percentage of frames that made it through our whole pipeline and produced a 6DOF estimate (i.e., not filtered along the way due to a detected error condition).

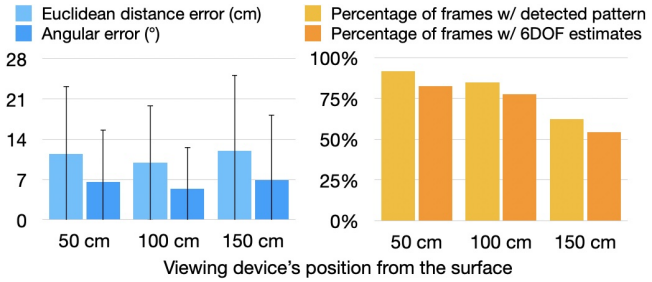


Figure 14: Study results broken out by the viewing phone's distance with respect to the surface. Left: Euclidean distance and angular error (all surface conditions combined). Right: The percentage of input frames where our pipeline detected the other device's projected pattern, along with the percentage of frames that made it through our whole pipeline and produced a 6DOF estimate (i.e., not filtered along the way due to a detected error condition).

the ground truth and predicted output using dynamic time warping (max correction window of ±5 frames, which equates to ±167 ms seconds at our system's 30 FPS), the mean distance error drops to 9.87 cm (SD=10.31) and the mean angular error drops to 5.93° (SD=8.49). This suggests our method is approximately 10% more accurate in practice. Nonetheless, we only report our uncorrected performance results below, offering a more conservative estimate of performance.

5.1 Effect of Surface

We evaluated our system on six typical surfaces: a gray table, a wooden table, a tile floor, a concrete sidewalk, a blue wall, and a white wall. Figure 12 summarizes our accuracy results. We did not find any significant differences in positional or rotational accuracy across our test surfaces. The percentage of detected pattern frames is around 85% for all surfaces, except concrete sidewalk (56.9%). In reviewing the collected data, we observed that the projected dots are more diffuse on the matte concrete surface vs. our other surfaces. The parameters and thresholds we set for contrast enhancement and dot detection did not perform as well, leading to a reduced percentage of frames with detected patterns. However, for patterns that are found, a similar fraction produces 6DOF estimates as our other surfaces, and the estimates are reasonably accurate.

5.2 Effect of Angle

The more oblique the viewing or projecting device is to a surface, the more the pattern becomes distorted (Figure 4). To understand the impact of this distortion on 6DOF estimation accuracy, we break out our results by angle in Figure 13.

For horizontal surfaces (tables and floors), a higher positional error of 12.83 cm (SD=16.22) is seen at 30° compared to other angles of attack; 8.91 cm (SD=9.47) at 60° and 9.92 cm (SD=10.16) at 90°. Orientation prediction has a similar trend: 9.00° (SD=15.23) at 30°, 5.42° (SD=7.37) at 60°, and 5.80 (SD=7.86) at 90°. In terms of the percentage of frames with detected patterns (Figure 13, right), 66.00% were found at 30°, while 60° and 90° angles had a mean of 85.14%. A similar trend is found with our vertical surface conditions.

5.3 Effect of Distance

We also analyzed the effect of distance on tracking accuracy, both for our handheld projecting device and our tripod-mounted viewing device. Starting first with the projecting device, there is a clear correlation in Figure 15 — positional error linearly increases as the device moves farther from the surface. This is likely due to even small errors in the P4P-predicted vector angle magnifying with longer rays. This error vs. distance effect can also be seen in the example point clouds in Figure 10.

Interestingly, we do not see a significant impact in accuracy as the viewing device gets farther away (Figure 14, left). Euclidean distance error is 11.44 cm (SD=11.77) at 50 cm, 9.88 cm (SD=10.01) at 100 cm, and 12.02 cm (SD=13.11) at 150 cm – a nominal increase in error. That said, we do observe that the percentage of frames with patterns detected does decrease with distance (Figure 14, right). By looking at our study data, we can see this is almost certainly due to the dots becoming less bright at longer distances, and thus harder to segment (especially darker and more diffuse surfaces). When we do detect the pattern, PatternTrack's 6DOF estimates are quite accurate.

6 Discussion

6.1 Comparison to Prior Methods

A unique advantage of PatternTrack is its ability to estimate the 6DOF position of another device — without utilizing any props or markers — using just a single frame of data (i.e., paired infrared camera and depth frames). SLAM-based methods, on the other hand, generally require users to pan the device around the local scene to collect enough visual and geometric data for a spatial registration to occur and further require either inter-device communication or a cloud service to mediate the pairing. In our experiences and from examples online, this registration process takes 5-10 seconds, which interrupts the user experience (see examples e.g., Unity AR Foundation Sample [76, 78], Apple's ARKit [73], and Microsoft Hololens [25, 77])

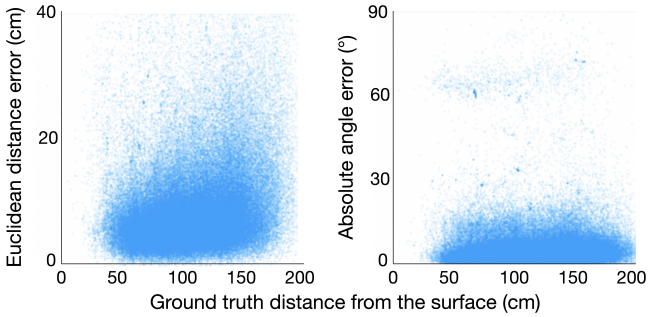


Figure 15: Left: Scatter plot of PatternTrack's Euclidean distance error vs. projecting phone distance from the surface. Right: Scatter plot of angular error vs. projecting phone distance from the surface.

In terms of spatial tracking accuracy, PatternTrack operates somewhere in between existing methods. RF methods, such as Bluetooth and UWB, offer roughly meter-scale accuracy (and generally only inter-device distance and not 6DOF device positions). By combing UWB with visual-inertial odometry, Cappella [52] was able to provide 0.9 m mean device tracking error. SynchronizAR (using UWB + SLAM, and requiring the room to be scanned first) [36] offered 0.2 m mean positional error (at an inter-device distance of 3 m distance). Using only SLAM, the Hololens was able to offer a mean tracking error of around 1.9 cm [37] (i.e., approximately 3.8 cm for two devices localizing one another, as the error would accumulate). We were not able to find tracking accuracy results for the Apple Vision Pro or Meta Quest 3, but they likely offer similar tracking performance. Tracko [38] (using BLE + acoustic time of flight) provides a 15.3 cm mean tracking error within a 1 m bubble. Printed fiducial markers, such as ArUco tags, tend to be very spatially accurate: roughly 0.5 - 1 cm error at 1 m ranges [22]. Structure light projection for use in projector-camera calibration offers similar precision.

6.2 Projection Surface Materials

One immediate limitation is that our technique does not work on glass, nor very dark or matte surfaces in the IR spectrum. This is simply due to the LiDAR pattern not being visible in our infrared image. There is no obvious way to support glass surfaces, but dark and/or rough surfaces might be enabled with a higher-quality camera sensor and higher camera exposure.

6.3 Environmental Lighting

As we are using active infrared projection, we are able to sidestep some common issues found in low illumination, e.g., tracking printed fiducial markers in a dark room. On the other hand, our technique struggles in bright infrared lighting conditions (e.g., outside on a sunny day), as the LiDAR pattern becomes washed out. A high-performance camera with a precise IR filter tuned specifically to the LiDAR's infrared frequency might extend the usable range of illumination. Fortunately, artificial indoor lighting (which generally does not have strong infrared components) does not significantly affect our pipeline. We note, for instance, that Apple's LiDAR-derived depth map appears to work well in both dark and very bright conditions, with minimal added noise. This leads us to believe a high-quality infrared image is available internally, enabled by carefully designed optics and image processing (tightly matched IR bandpass filters, application-specific auto exposure, etc.).

6.4 Projection Surface Geometry

Our current pipeline does not support operation on irregular surfaces (i.e., non-planar surfaces), such as furniture, people, and cluttered desks. This is because our pipeline makes the assumption the dot pattern lies in a plane. However, this is not an inherent limitation of our approach. As we have 3D scene information (from the device's built-in depth camera), we know the 3D location of every dot in the observed pattern. We can then solve a generalized Perspective-n-Point matching problem, which consists of identifying a subset of pattern rays and projection pose that optimally projects to the observed 3D dots.

6.5 Motion Blur

Motion blur does not appear to be a significant issue. In our study, with the projecting device in near-constant motion, we were able to detect the pattern in most frames, even out to our longest distance condition of 1.5 m from the surface. When analyzing failed frames, we found this is almost always due to a synchronization mismatch between the LiDAR emitter and our IR camera's rolling shutter, yielding incomplete patterns.

6.6 Sensing Range

As noted in Sections 2.1 and 4.2, we purposefully focused on typical co-located, multi-user collaboration distances, generally under 1.5 m. Our technical implementation and study design targeted this particularly valuable range (with the very longest inter-device ranges we captured being around 3 m). As can be seen in our results, accuracy at 1.5 m was the worst of our three viewing distance conditions. This is likely because even small errors in the transformation matrix (particularly orientation) get exaggerated at farther distances (e.g., 1° degree angular error matters much less at 50 cm than 150 cm). We note this is not unique to our method and other popular methods such as the 6DOF position derived from fiducial markers suffer from the same effect. To combat this issue, superior P4P estimates would have to be produced. A potential solution is to use a higher-resolution infrared camera, from which better dot centroids could be estimated. We note that our current infrared camera stream is only 640×360, and thus not particularly high resolution.

6.7 Computational Weight

As already discussed in Section 3.9, a present limitation of PatternTrack is its computational heft. Our brute-force rectangle generation and matching process is particularly expensive. For just one set of four dots (a candidate square), we solve P4P for four possible square orientations and translations on the pattern model image. A different pattern (not a regular grid) might be able to reduce the candidate set and also eliminate clearly wrong orientations and translations. Besides parallelization, another implementation path to boost the system speed is temporally storing patch location; based on the reasonable assumption that users will not move dramatically frame-to-frame, we can limit the candidate patches to specific regions and orientations, and then only occasionally activate a full search.

6.8 Overlapping Device Patterns

As mentioned in earlier sections, another limitation of our present prototype is the inability to tightly and rapidly control when the iPhone's LiDAR pattern is emitted. The best we can do with Apple's public APIs is to start and stop an ARKit session, which is inefficient and slow. With low-level (OEM) access, it is almost certainly possible to tightly control activations of the VCSEL emitter (independently of any depth-sensing operation), which can pulse in under 5 ns [29, 93]. Such a short duty cycle could allow for scores of co-located devices operating at 30 FPS to operate with few collisions probabilistically, and when collisions do rarely occur, devices can simply retransmit with a small random delay (a technique used extensively in shared medium multiple access communication approaches). Another option, also discussed above, would be for

proximate devices to synchronize their LiDAR patterns using e.g., Bluetooth.

One option we did not explore, but has promise, is to leverage the motion of patterns over time for segmentation [69, 88], as all of the dots for a single device move in a highly correlated manner.

Finally, we note that it is not possible to take advantage of motion-blurring techniques to mitigate overlapping patterns from multiple devices (e.g., ShakeSense [17], Maimone and Fuchs [47]). This is because the latter techniques have the effect of blurring other devices' patterns while retaining their own pattern (i.e., from a device's viewpoint only one pattern is visible: their own). PatternTrack devices, on the other hand, know their own (static) pattern and the data they need are the patterns of other devices.

6.9 Combining with Existing Techniques

We note that PatternTrack could be paired with other tracking techniques, including mature computer vision approaches (e.g., SLAM) or non-vision channels (e.g., UWB, Bluetooth, acoustics). For instance, Patterntrack could be used to rapidly initiate a shared AR experience, which then hands-off to a more conventional and accurate method like SLAM, leveraging the best of both methods. Another interesting avenue of future work would be to modulate the LiDAR patterns to carry data (akin to e.g., Visible Light Communication [12], LightAnchors [3], and InfoLED [94]). This would allow for projected patterns to encode not only the origin's 6DOF position, but also e.g., identity, user actions, and other information (similar to the interactions demonstrated in SideBySide [90] which projected data-bearing infrared fiducial markers). We also note some depth cameras use pseudo-random dot or speckle patterns (Kinect v1, Intel RealSense D456), which could be sufficiently unique so as to encode a user.

7 Conclusion

We have presented our work on PatternTrack, a new multi-device tracking method utilizing the infrared pattern projections of LiDAR sensors increasingly found in AR-capable devices, such as Apple's iPhone, iPad, and Vision Pro, as well as Meta's Quest 3. Our process needs only a single infrared+depth frame to make a 6DOF position estimate for other co-located devices, making interactions instantaneous without any registration step. This differentiates it from contemporary methods such as using a printed fiducial marker (requires a prop to be carried with the user) or fusion of multi-device SLAM data (requires pre-scanning the area). We believe the potentially lightwieght and instantaneous nature of our technique could help reduce barriers in creating future shared AR experiences. In our evaluation, testing a pair of co-located devices up to 2.6 m away from one another, on six common surfaces, we found a mean positional error of 11.02 cm and a mean rotational error of 6.81°. While considerable optimization and refinement remain to be done, we believe our initial result demonstrates the promise of this previously ignored infrared pattern for tracking purposes.

References

[1] Karan Ahuja, Mayank Goel, and Chris Harrison. 2020. BodySLAM: Opportunistic User Digitization in Multi-User AR/VR Experiences. In *Proceedings of the 2020 ACM Symposium on Spatial User Interaction* (Virtual Event, Canada) *(SUI '20)*. Association for Computing Machinery, New York, NY, USA, Article 16, 8 pages. https://doi.org/10.1145/3385959.3418452

[2] Karan Ahuja, Sven Mayer, Mayank Goel, and Chris Harrison. 2021. Pose-on-the-Go: Approximating User Pose with Smartphone Sensor Fusion and Inverse Kinematics. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 9, 12 pages. https://doi.org/10.1145/3411764.3445582

[3] Karan Ahuja, Sujeath Pareddy, Robert Xiao, Mayank Goel, and Chris Harrison. 2019. LightAnchors: Appropriating Point Lights for Spatially-Anchored Augmented Reality Interfaces. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 189–196. https://doi.org/10.1145/3332165.3347884

[4] Tim Althoff, Ryen W White, Eric Horvitz, et al. 2016. Influence of Pokémon Go on physical activity: study and implications. *Journal of medical Internet research* 18, 12 (2016), e6759.

[5] Gil Appel, Lauren Grewal, Rhonda Hadi, and Andrew T Stephen. 2020. The future of social media in marketing. *Journal of the Academy of Marketing Science* 48, 1 (2020), 79–95.

[6] Apple. 2023. AV Capture Device Type BuiltIn Wide Angle Camera. https://developer.apple.com/documentation/avfoundation/avcapturedevicetypebuiltinwideanglecamera

[7] Apple. 2023. Creating a Collaborative Session. https://developer.apple.com/documentation/arkit/creating_a_collaborative_session

[8] Apple. 2023. Creating a Multiuser AR Experience. https://developer.apple.com/documentation/arkit/creating_a_multiuser_ar_experience

[9] Apple. 2023. Nearby Interaction with UWB. https://developer.apple.com/nearby-interaction

[10] Apple. 2024. Capturing depth using the LiDAR camera. https://developer.apple.com/documentation/avfoundation/additional_data_capture/capturing_depth_using_the_lidar_camera

[11] Scope AR. 2023. https://www.scopear.com/solutions/worklink-platform

[12] Shlomi Arnon. 2015. *Visible light communication.* Cambridge University Press.

[13] Ján Bajana, Daniela Francia, Alfredo Liverani, and Martin Krajčovič. 2016. Mobile tracking system and optical tracking integration for mobile mixed reality. *International journal of computer applications in technology* 53, 1 (2016), 13–22.

[14] Riccardo Bovo, Daniele Giunchi, Muna Alebri, Anthony Steed, Enrico Costanza, and Thomas Heinis. 2022. Cone of Vision as a Behavioural Cue for VR Collaboration. *Proc. ACM Hum.-Comput. Interact.* 6, CSCW2, Article 502 (nov 2022), 27 pages. https://doi.org/10.1145/3555615

[15] M.S. Brown and W.B. Seales. 2002. A practical and flexible tiled display system. In *10th Pacific Conference on Computer Graphics and Applications, 2002. Proceedings.* 194–203. https://doi.org/10.1109/PCCGA.2002.1167859

[16] Frederik Brudy, Christian Holz, Roman Rädle, Chi-Jui Wu, Steven Houben, Clemens Nylandsted Klokmose, and Nicolai Marquardt. 2019. Cross-Device Taxonomy: Survey, Opportunities and Challenges of Interactions Spanning Across Multiple Devices. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–28. https://doi.org/10.1145/3290605.3300792

[17] D. Alex Butler, Shahram Izadi, Otmar Hilliges, David Molyneaux, Steve Hodges, and David Kim. 2012. Shake'n'sense: reducing interference for overlapping structured light depth cameras. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 1933–1936. https://doi.org/10.1145/2207676.2208335

[18] William AS Buxton. 1997. Living in Augmented Reality: Ubiquitous Media and Reactive Environments Redux. (1997).

[19] Li-Wei Chan, Hsiang-Tao Wu, Hui-Shan Kao, Ju-Chun Ko, Home-Ru Lin, Mike Y. Chen, Jane Hsu, and Yi-Ping Hung. 2010. Enabling beyond-surface interactions for interactive surface with an invisible projection. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology* (New York, New York, USA) *(UIST '10)*. Association for Computing Machinery, New York, NY, USA, 263–272. https://doi.org/10.1145/1866029.1866072

[20] T. Matthew Ciolek and Adam Kendon. 1980. Environment and the Spatial Arrangement of Conversational Encounters. *Sociological Inquiry* 50, 3/4 (1980), 237 – 271.

[21] CoSpaces. 2023. Make AR & VR in the classroom. https://www.cospaces.io/edu

[22] Gabriel M. Costa, Marcelo R. Petry, João G. Martins, and António Paulo G. M. Moreira. 2024. Assessment of Multiple Fiducial Marker Trackers on Hololens 2. *IEEE Access* 12 (2024), 14211–14226. https://doi.org/10.1109/ACCESS.2024.3356722

[23] Naa Amponsah Dodoo and Seounmi Youn. 2021. Snapping and chatting away: Consumer motivations for and outcomes of interacting with Snapchat AR ad lens. *Telematics and Informatics* 57 (2021), 101514.

[24] Starkey Duncan. 1983. Charles Goodwin, Conversational organization: Interaction between speakers and hearers. New York: Academic, 1981. Pp. xii+ 195. *Language in Society* 12, 1 (1983), 89–92.

[25] Mikkeline Elleby. 2023. Implementation of multi-user capabilties with the Microsoft Hololens 2. https://youtu.be/YWDDEXeGmyg?si=vpYYdjplMo1UJcYr. Accessed: 2024-12-10.

[26] Shenfeng Fei, Andrew M. Webb, Andruid Kerne, Yin Qu, and Ajit Jain. 2013. Peripheral array of tangible NFC tags: positioning portals for embodied trans-surface interaction. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces* (St. Andrews, Scotland, United Kingdom) *(ITS '13)*. Association for Computing Machinery, New York, NY, USA, 33–36. https://doi.org/10.1145/2512349.2512820

[27] Mark Fiala. 2005. ARTag, a fiducial marker system using digital techniques. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. IEEE, 590–596.

[28] George W Fitzmaurice. 1993. Situated information spaces and spatially aware palmtop computers. *Commun. ACM* 36, 7 (1993), 39–49.

[29] Rutronik Elektronische Bauelemente GmbH. 2022. VCSELs for ToF Applications - How VCSELs can be best put to use. https://www.rutronik.com/article/vcsels-for-tof-applications-how-vcsels-can-be-best-put-to-use

[30] Charles Goodwin. 1984. 10. Notes on story structure and the organization of participation. *Ann: Well-((throat clear))* 1 (1984), 4–0.

[31] Martin Habbecke and Leif Kobbelt. 2008. Laser brush: a flexible device for 3D reconstruction of indoor scenes. In *Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling* (Stony Brook, New York) *(SPM '08)*. Association for Computing Machinery, New York, NY, USA, 231–239. https://doi.org/10.1145/1364901.1364933

[32] Edward T Hall. 1966. The Hidden Dimension.

[33] Dai-In Han, Timothy Jung, and Alex Gibson. 2013. Dublin AR: implementing augmented reality in tourism. In *Information and Communication Technologies in Tourism 2014: Proceedings of the International Conference in Dublin, Ireland, January 21-24, 2014*. Springer, 511–523.

[34] Hooman Hedayati, Annika Muehlbradt, Daniel J Szafir, and Sean Andrist. 2020. Reform: Recognizing f-formations for social robots. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 11181–11188.

[35] Christian Holz, Frank Bentley, Karen Church, and Mitesh Patel. 2015. "I'm just on my phone and they're watching TV": Quantifying mobile device use while watching television. In *Proceedings of the ACM International Conference on Interactive Experiences for TV and Online Video* (Brussels, Belgium) *(TVX '15)*. Association for Computing Machinery, New York, NY, USA, 93–102. https://doi.org/10.1145/2745197.2745210

[36] Ke Huo, Tianyi Wang, Luis Paredes, Ana M. Villanueva, Yuanzhi Cao, and Karthik Ramani. 2018. SynchronizAR: Instant Synchronization for Spontaneous and Spatial Collaborations in Augmented Reality. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (Berlin, Germany) *(UIST '18)*. Association for Computing Machinery, New York, NY, USA, 19–30. https://doi.org/10.1145/3242587.3242595

[37] Patrick Hübner, Kate Clintworth, Qingyi Liu, Martin Weinmann, and Sven Wursthorn. 2020. Evaluation of HoloLens Tracking and Depth Sensing for Indoor Mapping Applications. *Sensors* 20, 4 (2020). https://doi.org/10.3390/s20041021

[38] Haojian Jin, Christian Holz, and Kasper Hornbæk. 2015. Tracko: Ad-hoc Mobile 3D Tracking Using Bluetooth Low Energy and Inaudible Signals for Cross-Device Interaction. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 147–156. https://doi.org/10.1145/2807442.2807475

[39] Haojian Jin, Cheng Xu, and Kent Lyons. 2015. Corona: Positioning Adjacent Device with Asymmetric Bluetooth Low Energy RSSI Distributions. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 175–179. https://doi.org/10.1145/2807442.2807485

[40] Adam Kendon. 2010. *Spacing and Orientation in Co-present Interaction*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1–15. https://doi.org/10.1007/978-3-642-12397-9_1

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

[42] Clemens Nylandsted Klokmose, Matthias Korn, and Henrik Blunck. 2014. WiFi proximity detection in mobile web applications. In *Proceedings of the 2014 ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Rome, Italy) *(EICS '14)*. Association for Computing Machinery, New York, NY, USA, 123–128. https://doi.org/10.1145/2607023.2610281

[43] Moritz Köhler, Shwetak N. Patel, Jay W. Summet, Erich P. Stuntebeck, and Gregory D. Abowd. 2007. TrackSense: Infrastructure Free Precise Indoor Positioning Using Projected Patterns. In *Pervasive Computing*, Anthony LaMarca, Marc Langheinrich, and Khai N. Truong (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 334–350.

[44] Ming Li and Leif Kobbelt. 2012. Dynamic tiling display: building an interactive display surface using multiple mobile devices. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia* (Ulm, Germany) *(MUM '12)*. Association for Computing Machinery, New York, NY, USA, Article 24, 4 pages. https://doi.org/10.1145/2406367.2406397

[45] Andrés Lucero, Jussi Holopainen, and Tero Jokela. 2011. Pass-them-around: collaborative use of mobile phones for photo sharing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 1787–1796. https://doi.org/10.1145/1978942.1979201

[46] Paul Luff and Christian Heath. 1998. Mobility in collaboration. In *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work* (Seattle, Washington, USA) *(CSCW '98)*. Association for Computing Machinery, New York, NY, USA, 305–314. https://doi.org/10.1145/289444.289505

[47] Andrew Maimone and Henry Fuchs. 2012. Reducing interference between multiple structured light depth sensors using motion. In *2012 IEEE Virtual Reality Workshops (VRW)*. 51–54. https://doi.org/10.1109/VR.2012.6180879

[48] Nicolai Marquardt, Robert Diaz-Marino, Sebastian Boring, and Saul Greenberg. 2011. The proximity toolkit: prototyping proxemic interactions in ubiquitous computing ecologies. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 315–326. https://doi.org/10.1145/2047196.2047238

[49] Nicolai Marquardt, Ken Hinckley, and Saul Greenberg. 2012. Cross-device interaction via micro-mobility and f-formations. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge, Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 13–22. https://doi.org/10.1145/2380116.2380121

[50] Rainer Mautz and Sebastian Tilch. 2011. Survey of optical indoor positioning systems. In *2011 International Conference on Indoor Positioning and Indoor Navigation*. 1–7. https://doi.org/10.1109/IPIN.2011.6071925

[51] Meta. 2024. Spatial Anchors Overview: Unity - Oculus Developer Center. https://developer.oculus.com/documentation/unity/unity-spatial-anchors-overview

[52] John Miller, Elahe Soltanaghai, Raewyn Duvall, Jeff Chen, Vikram Bhat, Nuno Pereira, and Anthony Rowe. 2022. Cappella: Establishing Multi-User Augmented Reality Sessions Using Inertial Estimates and Peer-to-Peer Ranging. In *2022 21st ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. 428–440. https://doi.org/10.1109/IPSN54338.2022.00041

[53] NDI. 2024. 3D Guidance: Quality OEM Tracking Solution - NDI. https://www.ndigital.com/electromagnetic-tracking-technology/3d-guidance

[54] Niantic. 2023. Pokémon GO. https://play.google.com/store/apps/details?id=com.nianticlabs.pokemongo&hl=en

[55] T. Ohshima, K. Satoh, H. Yamamoto, and H. Tamura. 1998. AR2Hockey: a case study of collaborative augmented reality. In *Proceedings. IEEE 1998 Virtual Reality Annual International Symposium (Cat. No.98CB36180)*. 268–275. https://doi.org/10.1109/VRAIS.1998.658505

[56] Takashi Ohta. 2008. Dynamically reconfigurable multi-display environment for CG contents. In *Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology* (Yokohama, Japan) *(ACE '08)*. Association for Computing Machinery, New York, NY, USA, 416. https://doi.org/10.1145/1501750.1501866

[57] OpenCV. 2024. CLAHE (Contrast Limited Adaptive Histogram Equalization). https://docs.opencv.org/4.x/d5/daf/tutorial_py_histogram_equalization.html

[58] OptiTrack. 2024. Motion capture system. https://optitrack.com

[59] Mikael Persson and Klas Nordberg. 2018. Lambda twist: An accurate fast robust perspective three point (p3p) solver. In *Proceedings of the European conference on computer vision (ECCV)*. 318–332.

[60] Danakorn Nincarean Eh Phon, Mohamad Bilal Ali, and Noor Dayana Abd Halim. 2014. Collaborative augmented reality in education: A review. In *2014 International Conference on Teaching and Learning in Computing and Engineering*. IEEE, 78–83.

[61] Polhemus. 2025. Motion tracking system. https://polhemus.com

[62] Voicu Popescu, G. Bahmutov, E. Sacks, and M. Mudure. 2006. The ModelCamera. *Graphical Models* 68, 5 (2006), 385–401. https://doi.org/10.1016/j.gmod.2006.05.002 Special Issue on the Vision, Video and Graphics Conference 2005.

[63] Voicu Popescu, Elisha Sacks, and Gleb Bahmotov. 2004. Interactive modeling from dense color and sparse depth. (2004).

[64] Milad Ramezani, Debaditya Acharya, Fuqiang Gu, and Kourosh Khoshelham. 2017. Indoor positioning by visual-inertial odometry. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 4 (2017), 371–376.

[65] Xukan Ran, Carter Slocum, Maria Gorlatova, and Jiasi Chen. 2019. ShareAR: Communication-Efficient Multi-User Mobile Augmented Reality. In *Proceedings of the 18th ACM Workshop on Hot Topics in Networks* (Princeton, NJ, USA) *(HotNets '19)*. Association for Computing Machinery, New York, NY, USA, 109–116. https://doi.org/10.1145/3365609.3365867

[66] R. Raskar, M.S. Brown, Ruigang Yang, Wei-Chao Chen, G. Welch, H. Towles, B. Scales, and H. Fuchs. 1999. Multi-projector displays using camera-based registration. In *Proceedings Visualization '99 (Cat. No.99CB37067)*. 161–522. https:

//doi.org/10.1109/VISUAL.1999.809883
[67] Ramesh Raskar, Jeroen van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao, and Clifton Forlines. 2006. iLamps: geometrically aware and self-configuring projectors. In *ACM SIGGRAPH 2006 Courses* (Boston, Massachusetts) *(SIGGRAPH '06)*. Association for Computing Machinery, New York, NY, USA, 7–es. https://doi.org/10.1145/1185657.1185802
[68] Ramesh Raskar, Jeroen van Baar, and Jin Xiang Chai. 2002. A low-cost projector mosaic with fast registration. In *Asian Conference on Computer Vision (ACCV)*, Vol. 3.
[69] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. 2024. SAM 2: Segment Anything in Images and Videos. arXiv:2408.00714 [cs.CV] https://arxiv.org/abs/2408.00714
[70] Jorge Rios-Martinez, Anne Spalanzani, and Christian Laugier. 2015. From proxemics theory to socially-aware navigation: A survey. *International Journal of Social Robotics* 7 (2015), 137–153.
[71] Tom Rodden and Gordon Blair. 1991. CSCW and distributed systems: The problem of control. In *Proceedings of the Second European Conference on Computer-Supported Cooperative Work ECSCW'91*. Springer, 49–64.
[72] Johannes Lutz Schönberger and Jan-Michael Frahm. 2016. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.
[73] Reality School. 2019. Create a Multiuser AR Experience RealityKit + Multipeer Connectivity. https://youtu.be/D1NhwmDUhYU?si=lZgjK_9lY6-1QkPq&t=2604
[74] V. Schwarz, A. Huber, and M. Tuchler. 2005. Accuracy of a commercial UWB 3D location/tracking system and its impact on LT application scenarios. In *2005 IEEE International Conference on Ultra-Wideband*. 599–603. https://doi.org/10.1109/ICU.2005.1570056
[75] Francesco Setti, Oswald Lanz, Roberta Ferrario, Vittorio Murino, and Marco Cristani. 2013. Multi-scale f-formation discovery for group detection. In *2013 IEEE International Conference on Image Processing*. 3547–3551. https://doi.org/10.1109/ICIP.2013.6738732
[76] si han ho. 2021. AR Foundation Samples: AR Collaboration. https://youtu.be/aRF0Gq1eCxU
[77] Kalloc Tech. 2019. HoloLens Multi-User Collaboration. https://youtu.be/uuea2eTgREw?si=9ewgZsJonKRa0iv6. Accessed: 2024-12-10.
[78] Unity Technologies. 2021. AR Foundation Samples. https://github.com/Unity-Technologies/arfoundation-samples/tree/4.1. Accessed: 2024-12-10.
[79] Mahdi Abbaspour Tehrani, M. Gopi, and Aditi Majumder. 2021. Automated Geometric Registration for Multi-Projector Displays on Arbitrary 3D Shapes Using Uncalibrated Devices. *IEEE Transactions on Visualization and Computer Graphics* 27, 4 (2021), 2265–2279. https://doi.org/10.1109/TVCG.2019.2950942
[80] Sebastian Tilch and Rainer Mautz. 2011. CLIPS proceedings. In *2011 International Conference on Indoor Positioning and Indoor Navigation*. 1–6. https://doi.org/10.1109/IPIN.2011.6071937
[81] Arda Ege Unlu and Robert Xiao. 2021. PAIR: Phone as an Augmented Immersive Reality Controller. In *Proceedings of the 27th ACM Symposium on Virtual Reality Software and Technology* (Osaka, Japan) *(VRST '21)*. Association for Computing Machinery, New York, NY, USA, Article 27, 6 pages. https://doi.org/10.1145/3489849.3489878
[82] S. Valin, A. Francu, H. Trefftz, and I. Marsic. 2001. Sharing viewpoints in collaborative virtual environments. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*. https://doi.org/10.1109/HICSS.2001.926213
[83] Vicon. 2024. Motion capture systems. https://www.vicon.com
[84] VIVE. 2024. VIVE Lighthouse. https://www.vive.com/us/accessory/base-station
[85] Daniel Wagner, Thomas Pintaric, Florian Ledermann, and Dieter Schmalstieg. 2005. Towards massively multi-user augmented reality on handheld devices. In *Pervasive Computing: Third International Conference, PERVASIVE 2005, Munich, Germany, May 8-13, 2005. Proceedings 3*. Springer, 208–219.
[86] Junyi Wang and Yue Qi. 2022. A multi-user collaborative AR system for industrial applications. *Sensors* 22, 4 (2022), 1319.
[87] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. 2024. DUSt3R: Geometric 3D Vision Made Easy. In *CVPR*.
[88] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. 2020. Towards Real-Time Multi-Object Tracking. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer International Publishing, Cham, 107–122.
[89] Reiner Wichert. 2002. Collaborative gaming in a mobile augmented reality environment. In *Proceedings of the Ibero-American Symposium in Computer Graphics*, Vol. 2002. 31–37.
[90] Karl D.D. Willis, Ivan Poupyrev, Scott E. Hudson, and Moshe Mahler. 2011. SideBySide: ad-hoc multi-user interaction with handheld projectors. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 431–440. https://doi.org/10.1145/2047196.2047254
[91] Andrew D. Wilson and Hrvoje Benko. 2014. CrossMotion: Fusing Device and Image Motion for User Identification, Tracking and Device Association. In *Proceedings of the 16th International Conference on Multimodal Interaction* (Istanbul, Turkey) *(ICMI '14)*. Association for Computing Machinery, New York, NY, USA, 216–223. https://doi.org/10.1145/2663204.2663270
[92] Robert Xiao, Chris Harrison, Karl D.D. Willis, Ivan Poupyrev, and Scott E. Hudson. 2013. Lumitrack: low cost, high precision, high speed tracking with projected m-sequences. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. Association for Computing Machinery, New York, NY, USA, 3–12. https://doi.org/10.1145/2501988.2502022
[93] Marcus Y. 2021. Lidar: Apple Lidar and DTOF analysis. https://4sense.medium.com/lidar-apple-lidar-and-dtof-analysis-cc18056ec41a
[94] Jackie (Junrui) Yang and James A. Landay. 2019. InfoLED: Augmenting LED Indicator Lights for Device Positioning and Communication. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 175–187. https://doi.org/10.1145/3332165.3347954
[95] Ka-Ping Yee. 2003. Peephole displays: pen interaction on spatially aware handheld computers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Ft. Lauderdale, Florida, USA) *(CHI '03)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/642611.642613
[96] Junko Yoshida. 2020. Breaking down iPad Pro 11's Lidar Scanner. https://www.eetimes.com/breaking-down-ipad-pro-11s-lidar-scanner
[97] Wenxiao Zhang, Bo Han, and Pan Hui. 2022. Sear: Scaling experiences in multi-user augmented reality. *IEEE Transactions on Visualization and Computer Graphics* 28, 5 (2022), 1982–1992.
[98] Jan Zizka, Alex Olwal, and Ramesh Raskar. 2011. SpeckleSense: fast, precise, low-cost and compact motion sensing using laser speckle. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (Santa Barbara, California, USA) *(UIST '11)*. Association for Computing Machinery, New York, NY, USA, 489–498. https://doi.org/10.1145/2047196.2047261