A Quest for Co-Located Mixed Reality: Aligning and Assessing SLAM Tracking for Same-Space Multi-User Experiences

Mark McGill
School of Computing Science
University of Glasgow
Scotland, UK
mark.mcgill@glasgow.ac.uk

Jan Gugenheimer
Télécom-Paris/LTCI/IP-Paris
France
jan.gugenheimer@telecom-paris.fr

Euan Freeman
School of Computing Science
University of Glasgow
Scotland, UK
euan.freeman@glasgow.ac.uk

ABSTRACT

Current solutions for creating co-located Mixed Reality (MR) experiences typically rely on platform-specific synchronisation of spatial anchors or Simultaneous Localisation and Mapping (SLAM) data across clients, often coupled to cloud services. This introduces significant costs (in development and deployment), constraints (with interoperability across platforms often limited), and privacy concerns. For practitioners, support is needed for creating platform-agnostic co-located MR experiences. This paper explores the utility of aligned SLAM solutions by 1) surveying approaches toward aligning disparate device coordinate spaces, formalizing their theoretical accuracy and limitations; 2) providing skeleton implementations for audience-based, small-scale and large-scale co-location using said alignment approaches; and 3) detailing how we can assess the accuracy and safety of 6DoF/SLAM tracking solutions for any arbitrary device and dynamic environment without the need for an expensive ground truth optical tracking, by using trilateration and a $30 laser distance meter. Through this, we hope to further democratise the creation of cross-platform co-located MR experiences.

CCS CONCEPTS

• **Human-centered computing** → **Virtual reality**; • **Computing methodologies** → **Mixed / augmented reality**;

KEYWORDS

Multi-User; Mixed Reality; Co-location; SLAM; VR; AR;

ACM Reference Format:
Mark McGill, Jan Gugenheimer, and Euan Freeman. 2020. A Quest for Co-Located Mixed Reality: Aligning and Assessing SLAM Tracking for Same-Space Multi-User Experiences. In *26th ACM Symposium on Virtual Reality Software and Technology (VRST '20), November 1–4, 2020, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3385956.3418968

1 INTRODUCTION

Mixed Reality (MR) devices are ever-present in our daily lives, from the ubiquitous Android (ARCore) and iOS (ARKit) smartphones, to the commodity VR and AR headsets (e.g. Oculus Quest, Hololens 2) that are seeing steady uptake by both consumers and industry. With respect to shared and social experiences, there has been an understandable emphasis on at-a-distance usage, such as VR multiplayer gaming (e.g. [32]) and social VR (e.g. Mozilla Hubs [28], Facebook Horizon [12]). However, MR devices also have the potential to create highly engaging co-located experiences [17], in theory being able to augment, or even entirely supplant, reality for groups of people in the same space. Recently, there has been a concerted push by MR platform providers to support co-location. This has been driven by the ubiquity of 6DoF "inside-out" tracking in MR devices, relying on visual-inertial odometry, using camera-based Simultaneous Localisation and Mapping (SLAM) and Inertial Measurement Unit (IMU) sensing to track the real-time position and orientation of a device relative to the real world environment. These solutions have become increasingly robust and (seemingly) reliable e.g. being the defining feature of the Oculus Quest VR headset, enabling both positional tracking and the safety-critical "guardian" boundary.

Given such 6DoF tracking, the creation of multi-user co-located experiences requires only that we can determine a transformation between each MR device coordinate space and our real-world coordinate space [9]. Recent efforts by Microsoft [3, 31], Google [27], Apple [26] and Oculus [33] have focused on synchronised (often cloud-based) spatial anchors, allowing for common reference points across clients, or synchronised SLAM data [33] to fully align device coordinate spaces with reality. However, these solutions are often "black boxes" - closed source solutions, introducing tight coupling to specific MR SDKs, often relying on internet-dependent cloud-based APIs that introduce concerns regarding costs and privacy, with interoperability across platforms often limited by design as a means of creating a 'walled garden' for a given device ecosystem. Conversely, whilst co-located experiences have been repeatedly examined in research without relying on such dependencies [9, 16, 20, 23], the methods by which aligned experiences (from seated audiences to arena-scale co-location) were facilitated have never been sufficiently documented to enable replication.

The aim of this paper is to better equip practitioners in creating basic co-located MR experiences, and remove the necessity of depending upon specific platforms, freeing them to use any combination of available SLAM-tracked devices that suit both their specific needs, and the context they are deployed in. We do this through an exploration of aligned SLAM approaches, where we determine transformations to pre-determined points in reality, allowing disparate SLAM devices to quickly align their coordinate systems. We do so in three ways: 1) we describe, through pseudo-code, how to accurately align devices to both small-scale (using one-point alignment) and large-scale (using two-point alignment) spaces, and formalize the accuracy and limitations of these approaches; 2) we

VRST '20, November 1–4, 2020, Virtual Event, Canada

provide example scenes with skeleton implementations using these alignment techniques to enable seated audience, small-scale and large-scale co-location scenarios, demonstrating the feasibility of co-location across generic 6DoF XR devices and ARCore smartphones; 3) we detail a novel approach, using two-point alignment, allowing practitioners to assess the accuracy and precision of 6DoF tracking for any given space/context without the necessity of external optical tracking, effectively enabling practitioners to determine whether a given combination of an MR device and physical environment (e.g. with moving people, changing lighting etc.) can safely support co-location, using only a $30 laser distance meter.

Whilst originally motivated by our own practical needs working with artists, our contributions will help practitioners and researchers to create co-located MR experiences right now, with no platform-specific dependencies. We provide all our implementations to the community via Github and hope to better inform the community regarding the potential for democratized, low-cost, cross-platform co-located mixed reality, supporting researchers, students and artists in creating shared and social MR experiences.

2 BACKGROUND

Consumer MR devices have often presented a barrier to co-located interaction, resulting in isolation of the wearer, and exclusion of non-headset users [16]. However, their merit in co-location has been well explored. For collaboration, co-located MR has been deployed in a variety of contexts, from construction and engineering, education, entertainment and gaming, tourism and more, accounting for approximately 40% of collaborative MR research from 2013–2018 [2], with notable benefits in terms of cognitive load, cooperation, and awareness. For play, research into co-located augmented playspaces (CAPs) [34] has established benefits in terms of stimulating physically active behaviour, social interactions, cognitive development in children, and fundamentally in enabling unique, joyful experiences. And in art, co-located MR performances such as CAVRN [20], Holojam [15], and CAVE [23] have pushed the boundaries in terms of audience experience, with "the adaptation of both cinematic and theatrical elements offer[ing] a unique set of affordances for content designers and producers" [23]. Such experiences might incorporate multiple types of MR devices and displays (the utility of which was particularly emphasised by ShareVR [17]) and multiple user roles, from passers-by, to spectators, to headset users [37].

2.1 Visual-Inertial SLAM Across Devices

SLAM tracking [11] generates knowledge of the real-world environment using optical data which is then fused with IMU data to allow for high sample rate positional tracking. For MR headsets and devices, this is typically referred to as "inside-out" positional tracking, and such spatial positioning has become a commonplace feature of VR and AR devices, necessary to create exocentric, 6DoF tracked MR experiences. Such solutions can operate as relative positional tracking (e.g., relying on optical flow and IMU data), but typically also retain a spatial memory of the environment (e.g., through stored point clouds of environmental features), meaning that positioning can be absolute if there is sufficient knowledge of the environment. Note that tracking is only absolute with respect to each individual device, being internally consistent – the inter-device coordinate spaces are inevitably different (based on the starting point/alignment, the state of the IMU, etc.). To create co-located experiences with multiple devices we then need to be able to synchronize their spatial frames/align their coordinate spaces.

2.1.1 Alignment using Features, GNSS. SynchronizAR [22] demonstrated how ad-hoc co-located experiences could be constructed spontaneously and implicitly. They used Ultra-Wide Bandwidth (UWB) modules attached to each MR device, providing distance measurements between all devices in the co-located session, which in turn allowed for the positions (relative to each other) of the MR devices to be resolved. Once synchronisation was complete, they relied on the in-built SLAM tracking for providing subsequent position updates as users moved throughout the space. This approach was both highly novel and effective, however this did require bespoke hardware attached to each MR device. Discounting additional hardware, common tracked features can also be used for alignment. For example, ArUco/fiducial markers or LightAnchors [1] with parallel positions in the virtual scene are sufficient for a device to align it's coordinate space to said marker to an extent. Every major AR platform has some mechanism for supporting this implicitly without markers, typically through *spatial anchors*. These define a point that can be tracked across devices and time through identification of features in the physical environment. For example the Microsoft Mixed Reality SDK supports Spatial Anchors directly [31] which can be shared across devices supporting the major AR toolkits (ARCore, ARKit, Microsoft) using *Azure Spatial Anchors* [3]. This situation is mirrored on ARCore with *Cloud Anchors* [27] and iOS with the *ARWorldMap* [26]. Given visibility of a shared spatial anchor, we can align the individual coordinate space of each MR device to these anchors. However inaccuracy in identifying and tracking a singular marker or feature can introduce significant errors in alignment as Defanti noted: "simple calculations yield that even 5deg of error will cause objects to have almost 10cm of error per 1m they are positioned away from the frame origin" [9].

Indoors, a plethora of research has examined transposing trilateration/triangulation approaches for e.g., WiFi, RADAR [5], and other technologies (summarized by [38]). And for outdoor experiences, we can incorporate Global Navigation Satellite System (GNSS) data, capable of providing absolute positioning to anywhere from multimeter to (soon) deci-centimeter level accuracy [6]. However, such solutions have their own caveats e.g. in terms of signal availability, accuracy and reliability, although work by companies such as Niantic on their "world platform" [21] is rapidly bridging this sensor fusion gap of external triangulation + SLAM data.

2.1.2 Synchronised SLAM. Alternatively, there is the synchronisation of SLAM data to consider. Instead of tracking one or more features and aligning to those, instead devices can share [29], and even collaboratively map [35], the environment such that they all share the same coordinate space and absolute positioning within this environment. Perhaps the most relevant example of the synchronised SLAM approach was demonstrated by Oculus in 2018 [33]. They showed an arena-scale co-located VR game, where synchronization appeared to be achieved through sharing a pre-captured map of the environment that all headsets could refer to. This allowed the inside-out tracking to determine the absolute position of the headset relative to the environment, without any necessary

re-alignment or transformation to match other device's coordinate spaces. This also had the benefit of being able to share pre-defined safety boundaries (the Oculus 'Guardian' system) for the arena environment. This demonstration prompted a variety of developers to work on the creation of such experiences, with it being noted by Auxietre that "we held full-scale playtests in an 'ideal' setting of 30x15m (50x100feet) with 8 simultaneous users and it worked nicely on development hardware" [4].

Synchronised SLAM and shared spatial references are the ultimate end-point of ubiquitous co-located MR. However, they do have a number of caveats to be considered. For synchronised SLAM, [29] noted in particular issues regarding end-to-end latency in discovery and alignment, bandwidth usage (particularly at the initial synchronisation of a map), and processing requirements (particularly if offloading elements of the map processing to the cloud). Across both solutions there are common caveats in terms of:

Internet access They often require an internet connection with a dependency to cloud infrastructure (although P2P synchronisation is sometimes supported, e.g., with *ARWorldMap*), introducing a potentially critical failure point for public deployments;

Privacy They require the transmission of summary data regarding the device (e.g. GNSS position, device orientation, calculated hashes or sparse point clouds of camera data) either P2P or to the cloud. The sharing of this data introduces concerns regarding privacy in particular - your real-world environment may be mapped and made accessible to applications on other devices;

Platform-Specific Dependencies There is no one common SDK that supports all devices, restricting development to "Walled Gardens" of specific devices on supported platforms.

This latter point is perhaps most prescient. The aforementioned Oculus Co-location API has yet to be released, with developers noting the lack of communication from Oculus regarding public access to this API restricting the development of co-located VR [4]. Indeed, Oculus restrict the capability of developers to create co-located experiences, with it being noted [18] that the Oculus for Business Enterprise Use Agreement contains the restriction that "Unless separately approved in writing by Oculus, you will not... modify the tracking functionality (including the implementation of any custom co-location functionality) on your Software". Even when Oculus release their co-location SDK, there is still the question of how MR headsets and devices from other manufacturers might be integrated into shared experiences—if at all. With reference to the Oculus SDK specifically, Heaney [19] noted that co-location code resides as part of the Oculus Platform SDK, requiring approval for developer access and leveraging the Oculus store and servers. Moreover, for privacy reasons, the cameras on the Oculus Quest cannot yet be accessed by developers, in turn hindering support for other platforms that might facilitate co-location such as ARCore.

In fairness, this situation is by no means unique to Oculus e.g., the Apple ARWorldMap is supported only on iOS devices, whilst Azure spatial anchors aren't supported on Magic Leap devices, and so on. Indeed, such incompatibilities are likely to be exacerbated, at least in the short term, as companies compete for dominance of the MR marketplace, and common standards emerge. Whilst companies are exploring general purpose, cross-platform alignment solutions (e.g., Spatial.io [30]) there are likely to be many cases where bespoke, low-cost, platform agnostic solutions would prove invaluable.

2.2 Assessing The Accuracy of SLAM Tracking

Given the reliance on optical tracking, the localisation accuracy (i.e., that the device determines its position in reality correctly) and precision (i.e., that the position is consistent and repeatable) of SLAM solutions can be highly dependant on the environment they are used within, and specifically how that environment physically changes over time (e.g. the presence of moving others). Feigl *et al.* [13] assessed the localisation accuracy of ARCore, ARKit and Hololens devices, finding that "out of the box, these AR systems are far from useful even for normal motion behaviour", with an average error of approximately 17 m per 120 m when assessed in a large-scale industry environment (60 m+ traversals). And Duque [10] assessed the Oculus Rift CV1 outside-in tracking, finding distance error of approximately 1.7 cm. Feigl *et al.* in particular found that the mean absolute position error was deoendent on the number of optically unique features in an environment, the presence of RGB-Depth sensing on the device, and dynamic, prolonged occlusion of the environment. Consequently, determining whether a given SLAM-tracked device will work for a particular environment/use case is difficult to answer without some form of assessment in-situ. Problematically, common to both these papers was the use of external optical tracking as a ground truth for benchmarking (e.g., ARTTRACK cameras in [13], outside-in tracking cameras in [10]). This significantly increases the cost and complexity of in-situ assessments in different spaces, with optical tracking solutions capable of tracking devices over large spaces costing significant sums and requiring the installation of hardware infrastructure.

3 BUILDING CO-LOCATED MR EXPERIENCES

Our focus is on facilitating platform-agnostic co-located MR experiences using inside-out positionally tracked MR devices (assumed to have a robust spatial memory). In an evaluation-by-demonstration [24], we walk the reader through two approaches for aligning a MR device with a real-world space, aligning to a single known point in reality (requiring known position and orientation, suited to small roomscale co-location), or two known points (requiring position only, suited to large roomscale co-location). For each approach, we detail the potential accuracy and pitfalls in application, and provide demonstrator scenes exploring real-world application, across seated audiences, and small-/large-scale co-location, that work for both AR smartphones (ARCore) and generic XR headsets, with the full Unity project provided at github.com/mark-mcg/VRST-20-Aligned-SLAM-Exemplars. For all scenes, we provide a basic client-server network implementation using Mirror [25].

3.1 Aligning to a single known point

The most basic means of alignment is to perform a *one-point alignment*, a one-shot transformation from the device's current position+orientation in reality to a specified position+orientation in the virtual coordinate space which has a congruent, equivalent position+orientation in reality. This approach has typically been seen in projects such as CAVE and CAVRN [20, 23], where each audience seat in reality has an equivalent position noted in virtuality,

such that the MR device can be aligned with said point, giving the audience member the equivalent view in MR as in reality, and such a transformation can be generated trivially:

Listing 1: Alignment to known point in VR

```
// Align orientation by rotating the device by the angular
    ↪ difference
Quaternion Rotation = Quaternion.Euler (new Vector3(0, target.transform.
    ↪ eulerAngles.y − device.transform.eulerAngles.y, 0) );
// Rotate the device position around origin to match new orientation
Vector3 Position = device.transform.position.RotateAroundPivot(Vector3.zero,
    ↪ Rotation);
// Calculate the difference between the rotated and target positions
Vector3 Translation = target.transform.position − Position;
// ... then apply rotation and translation to parent of tracked MR
    ↪ device ...
```

This alignment approach is simple but effective - you need only define a point in virtuality ('target'), and an equivalent point in reality that your device can track; e.g., aligning a headset or controller with this point, or tracking a fiducial marker/QR code/spatial anchor at this point, so that we have a position and orientation in the virtual coordinate space that we can align with the known position/orientation of the 'target' in reality. Multiple devices can be aligned in this way, meaning that, regardless of differences in their individual coordinate spaces, the eventual position of the devices will be aligned, allowing for co-located MR.

3.1.1 Practicality of Approach[1]. The effectiveness of aligning to a single known point is highly dependent on how accurately we can determine the virtual equivalent of our 'target' in reality. Consider multiple VR users that each stand at the same noted point in reality, and point their headsets toward a pre-determined feature in the room. Each pulls the trigger of their VR controller, enacting a one-point alignment. Once finished, each device will be theoretically aligned for co-located interaction. However, in practice, each user will vary in terms of the precision of positioning and orientation of the device, and these errors can have significant impact on the perceived alignment [9]. Consider a circular play area with radius R, with an alignment point in reality at the centre of the play area Q_i. If the user attempts to align the device with Q_i but is slightly off on the position (P_i) and angle (θ_1), and then walks to the edge of the play area (distance R) along the x axis, then their expected position Q_e and actual position P_a would be separated by a distance of E_p, our positional error ($E_p = d(Q_e, P_a)$), as can be seen in Figure 1. To calculate this, we need only find the position of point P_a, which can be determined as:

$$P_a = P_i + (R.cos(\theta_1), R.sin(\theta_1)) \quad (1)$$

For simplicity, if we consider the impact of the angular error alone for different radii of play areas, assuming alignment occurs at the center of the play area, we can see that the resulting position error by the time we reach the edge of the play area (i.e. having travelled by R) increases linearly with angle (see Figure 2). For small play areas, there is a reasonable margin for error in calibration here, with $\pm 4.5°$ still allowing for positional error within $\pm 0.1m$ for a $1.5m$ radius play area. However, as the play area size increases, this margin for error becomes increasingly tight, requiring precise initial angular alignment. There is a clear trade off for this approach - it is quick to enact, but not well suited to large play areas.

3.1.2 Exemplar 1: Seated Audiences. The most basic co-located application of this approach is in the creation of seated audience experiences - alignment with reality is achieved merely by ensuring that audience seat locations in MR mirror their physical arrangement and positions precisely in reality. In exemplar scene (1 `Audiences.unity`), we provide a basic mock-up of audience seating for an immersive co-located VR experience such as CAVE [23]. We have created 6 GameObjects denoting 6 alignment points, one for each seat (see Figure 3), each with attached `OnePointAlignment` components. In every example in this paper, the alignment components extend a base `TransformationToReality` component which specifies what device or marker should be treated as the equivalent point in VR when alignment is enacted (e.g., 3/6DoF headset, tracked object such as a 6DoF controller, or tracked marker such as a QR code), and whether we should align to this point on xyz, or just xz (e.g. for aligning a headset whilst retaining the height of the user relative to others). By default, this is taken as the available

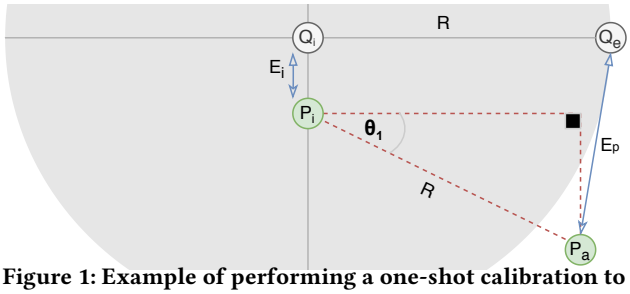

Figure 1: Example of performing a one-shot calibration to a known point Q_i, with an initial positional error of E_i. If the user moves by R along the x axis, they would be expected to be at point Q_e in reality. However, given potential angular mis-alignment of θ_1, their resultant actual position would be estimated to be P_a, with an overall error of E_p.

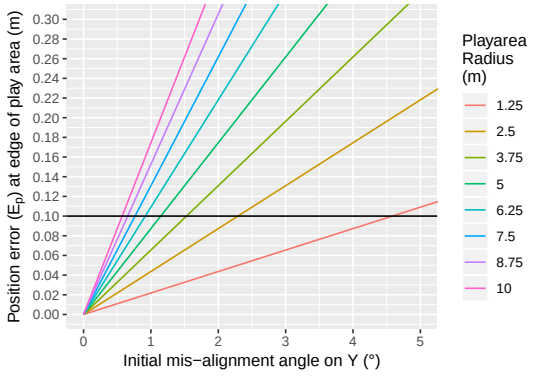

Figure 2: Expected position error E_p solely as a result of angular mis-alignment (i.e. not including E_i), for varying sizes of play area and angles of mis-alignment. An error of $\pm 0.1m$ is highlighted as a reasonable maximum allowable error.

[1]**N.B.** For all discussions regarding accuracy hereafter, we assume a circular play area boundary to simplify position error calculations - for any non-circular play area, refer to the largest enclosing circular play area for the worst case error. For angular error we refer to yaw error on Y - orientation differences on X/Z between devices are assumed to be corrected by IMU sensor fusion using the direction of gravity.

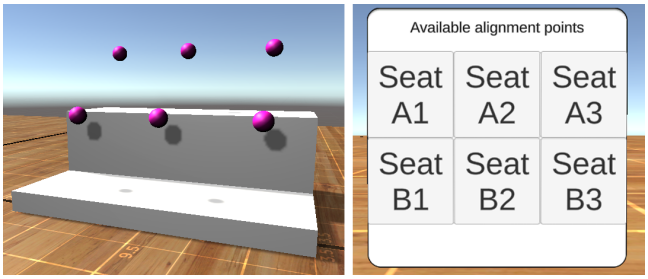

Figure 3: Left: Each sphere represents an audience member's seat alignment position in VR with a congruent point in reality. Right: UI for allowing users to select their seat to align their headset manually.

Figure 4: Shows an Oculus Quest and Android ARCore device both aligned using one-point alignment, captured from the perspective of the ARCore smartphone device with pass-through video enabled. The pink spheres represent the center of gravity of the headset and controllers, with a head and hand models attached to those points in the XR player prefab. See video figure for footage this picture was taken from.

MR headset position. This scenario is unique in that it works for both 3DoF and 6DoF headsets, thus SLAM tracking is not strictly required (provided the audience members remain seated).

This scene is setup to demonstrate alignment using the XR headset, meaning that the headset needs to be held, or worn, such that it is in the same comparable point in reality as it will be calibrated to in VR. We can indicate which location we are aligning to in a number of ways e.g. pre-determined based on a config file on the device, selected by the user via a UI (as shown in Figure 3), or conveyed remotely by an operator, and then enact the alignment procedure when the headset is in said position. The bundled UI queries an `AlignedPointManager` class and presents all available `TransformationToReality` components as options to align to. The result is that the headset will be calibrated such that, when the user wears the headset, they will perceive themselves to be at approximately the same location and orientation in reality as they are in virtuality.

The biggest issue with such a setup is if the alignment procedure is not followed correctly or if the incorrect seat is chosen. Such a setup will be relatively forgiving of minor mis-alignments, given that users will be seated throughout the experience, however consider that the user selects seat A1 instead of seat B3. Their view of the scene would thus be markedly different than their position in reality. Equally, if the user performs the calibration by aligning an MR device directly, and the device is in the wrong orientation, they would find that the MR world would be offset by the difference between the expected and actual orientation at alignment. CAVE [23] appears to have used an operator-driven approach, and it seems likely that the headsets were placed in their pre-determined seat positions for calibration prior to audience arrival, or that audience members were asked at the start to look straight ahead prior to calibration. If the headset has an available front-mounted camera feed (not possible on the Oculus Quest, for example), alternatively placing a fiducial marker/QR code on the space in front of the real-world seat would allow for headsets to auto calibrate themselves, however any inaccuracy in detecting the orientation/position of the marker would again impact alignment. These trade-offs would need to be thought through depending on the scale and longevity of the expected deployment.

3.1.3 Exemplar 2: Small-Scale Co-Location. This exemplar (`Small Scale Colocation.unity`) is a facsimile for co-located experiences that are roomscale in scope; i.e., modest living-rooms or offices, where a small number of people might operate within restricted confines (e.g., around $3 * 3m$) for a relatively ad-hoc experience. Again, the exemplar scene is setup for directly aligning the headset with a known point, this time performing a one-point alignment with a point in the center of the available play area (ensuring best accuracy as we move toward the center of the play area), facing/aligned with a known landmark for orientation, with alignment occurring either through on-screen UI (for smartphones) or XR primary trigger input. The end result is that we can have n devices sharing an aligned play space, as seen in Figure 4. Again, we could swap out direct alignment of the headset with detecting a visual marker, if supported, but this could introduce further inaccuracy based on errors in estimating the position/orientation of said marker.

As with the audience example, because we rely on a one-point alignment, accuracy again depends on the alignment being conducted appropriately by each user. And, as detailed previously, accuracy at the edges of the play area will be determined by the quality of this initial calibration. We suggest that this approach be used for settings where a quick, rough ad-hoc alignment is required because of these issues. For example, aligning multiple smartphones and AR headsets to interact with shared virtual content, or providing smartphone users with a perspective correct view of a VR user's actions. In both cases, there is little in the way of roomscale movement, nor the possibility of collisions between blinded VR users, so issues regarding the accuracy of alignment are minimised.

3.2 Aligning to two known points

Given the degradation in accuracy for one-point alignment as we move away from the alignment point, for larger co-located play areas, we need to consider how we might provide an alignment approach that is more robust to the error in positioning/aligning the MR device (e.g., headset, controller, identified QR code or spatial anchor) to a known point in reality. DeFanti [7] considered that the angular error was the predominant issue with the latter approach, and proposed aligning the position only to two known

points in reality to effectively remove the angular error from consideration. They noted that "if a user can accurately place the headset within 1cm of each of the fixed points [at opposite ends of the play area], then there will be no more than 1cm of error throughout the tracked space". Consider a circular play space of radius R, with the headset positioned with a degree of error at each of the two alignment points, P1 and P2. If the points chosen are opposing points at the edge of this play area, then the positional error will be, at worst, equivalent to the error at the nearest recorded alignment, with the maximum angular misalignment on y being $\theta_2 = \text{atan2}(P2.y - P1.y / P2.x - P1.x)$. Performing such an alignment is again straightforward (see Listing 2), requiring two known points in reality, with the position of the MR device (or controller, marker etc) recorded when the device is at each of said points.

Listing 2: Alignment to two known points, Q1 and Q2, in MR

```
// Calculate the rotation on the y axis for alignment
Quaternion Rotation = Quaternion.Euler(0, Quaternion.FromToRotation(P2.
    ↪ transform.position − P1.transform.position, Q2.transform.position − Q1.
    ↪ transform.position).eulerAngles.y, 0);

// Rotate the recorded device points around origin to align
    ↪ orientation
Vector3 P1PositionR = P1Position.RotateAroundPivot(Vector3.zero, rotation);
Vector3 P2PositionR = Q2Position.RotateAroundPivot(Vector3.zero, rotation);

// Find the translation between the centroids of Q1/Q2 and P1R/P2R
Vector3 Translation = ((Q1Position + Q2Position) / 2) − ((P1PositionR + P2PositionR
    ↪ ) / 2);

// ... then apply rotation and translation to parent of tracked MR
    ↪ device ...
```

Practicality of Approach. This alignment approach requires more effort in calibration, requiring that for each device we align a tracked object (e.g., headset, controller, marker) to two points in our play area. However, once this alignment has been performed, this approach will provide greater accuracy than the one-point approach, and is relatively fool-proof in terms of conducting the alignment thanks to the orientation of the device/alignment points being of no consequence. With this, accuracy at the boundaries is dictated by how accurately we can align to each of our points, and how far apart those points are. The latter point is, however, potentially problematic for very large play areas (e.g., arena scale), requiring that each device be calibrated to points at the far reaches of the play area at least once (assuming we can store and retrieve this calibration for the given real world location, which may not always be possible). For practicality, let us assume we cannot store and retrieve this calibration, and wish to calibrate to two points closer together than the maximum extents of our play area and are willing to sacrifice accuracy to some extent e.g., using two points at a denoted location where a headset is calibrated and handed to a user, at the edge of the play area (see Figure 5).

Beyond the encircling boundary of $Q1$ and $Q2$, the error E_p will scale linearly as we move further from the centroid C (imagine drawing a line between P1 and P2, and carrying that line further - this line represents our position including error). Consequently, we can approximate the positional error E_b at the furthest boundary from C in our play space, x, through a ratio of the distance $d(x, C) = R + d(C, O)$ and $d(Q1, C)$:

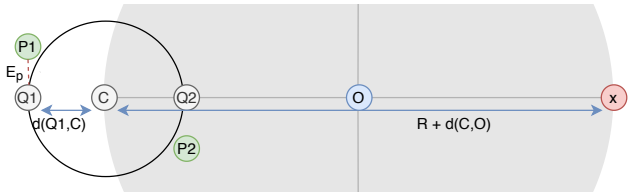

Figure 5: If we calibrate to two known points Q1 and Q2 of an arbitrary distance apart, with an accuracy of E_p, placed at the very edge of our play area, this shows the approximate distance to the furthermost point in the play area.

$$E_b = E_p \frac{R + d(C, O)}{d(Q1, C)} \tag{2}$$

In practice however, it would be useful to plug in the size of the play area, where the alignment centroid will be (i.e., where we will be performing the alignment), and what our required accuracy at the furthest reaches of the play area might be, to determine the minimum required distance between our calibration points:

$$d(Q1, C) = \frac{E_p}{E_b}(R + d(C, O)) \tag{3}$$

So for a play area of 5 m in radius, with a measuring accuracy of $\pm 0.01m$ and a tolerance of $\pm 0.1m$ at the edges of the play area, with calibration performed at the edge of the play area (5 m from center), we would need to measure two points with a centroid at the edge of the play area separated by $2 * (0.01/0.1) * (5 + 5)) = 2m$. Approximations for expected accuracy assuming the calibration always occurs at the edge of the play area can be seen in Figure 6. Given that we can predict the accuracy at any point in the play area, we could also incorporate this knowledge into the design of our co-located experience; for example, making safety boundaries around users larger as they reach the edge of the play area, or adjusting targeting to compensate.

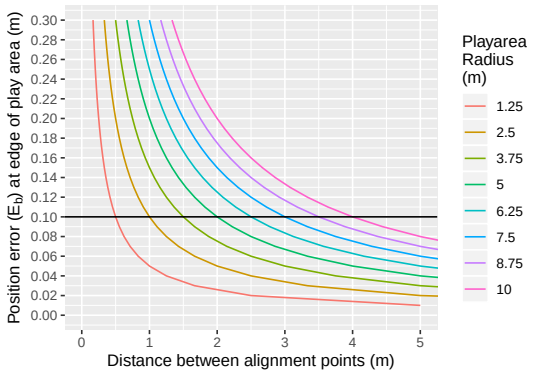

Figure 6: Plot of distance between alignment points for two-point alignment approach for play areas from 5m to 10m, against the expected accuracy of alignment when the device is at the edge of the play area (E_b), with $0.1m$ worst case accuracy highlighted. A $\pm 0.1m$ position accuracy (E_p) is assumed.

3.2.1 Exemplar 3: Large Scale Co-location. This exemplar (see `3 Large Scale Colocation.unity`) demonstrates the use of a two-point alignment for enabling large-scale multi-user co-location. For this, we need some pre-existing knowledge of the environment, specifically the distance between our two alignment points in reality, such that we can place our virtual alignment points the same distance apart. Said distance should be determined by the practitioner based on our previous accuracy guidance.

For such a scenario, errors in tracking, and accidental/inadvertent resets of the user's view, can become a safety-critical concern. Consider that the user accidentally holds the Oculus menu button, and resets their orientation and position. Or consider that the MR device SLAM tracking may determine it is in an entirely different position in reality to where it was the previous frame, and instantaneously teleport the MR user to the seemingly new 'correct' position. In both cases, our position in virtuality is no longer correctly aligned with reality. There are two mechanisms by which we can deal with such errors. Firstly, we can detect movements that are deemed physically impossible, based on the expected movements of the devices. `TrackingFailureDetector` provides an example of this, where we can act on such an event by, e.g., requesting the user take off their headset and return to the start/perform a re-calibration. Secondly, if the device exposes some knowledge of the position/alignment of the real-world space such as a safety boundary, we can utilize this knowledge to recover from any significant changes in the device's virtual coordinate space. We can retrieve the boundary positions, find a transformation from our current boundary positions to the boundary positions when we performed our alignment, and apply both this transformation and our previous transformation to once more arrive at an experience that is aligned correctly relative to reality (again, first postulated by DeFanti [8]). In this way, we can use aligned SLAM tracking safely to enable co-located experiences bounded only by the spatial memory and capability of the tracking on our MR devices. Moreover, the burden of performing an alignment can be lessened for long term deployments in the same space.

4 ASSESSING SLAM ROBUSTNESS

Finally, we detail a novel low-cost approach, using two-point alignment, toward assessing the accuracy of a SLAM-tracked device in any given environment/context, requiring only a $30 laser range finder. This step is crucial for establishing whether a given SLAM device will operate robustly enough for alignment approaches to be viable, and more broadly empowers practitioners to readily assess the potential safety and effectiveness of SLAM devices in any environment (e.g. large spaces, in changing lighting conditions, with moving crowds etc) without necessitating an expensive optical ground truth. In this way, this paper provides all the necessary support for practitioners and researchers to create and deploy their own cross-platform co-located MR experiences.

This latter point is, however, highly prescient — the accuracy of our alignment techniques is dependent both on the application of each technique, and also the underlying robustness of the SLAM tracking implementation. We could trivially perform a two-point alignment with $0.01m$ worst case theoretical error at the edges of the play area, however the error in reality is also contingent on the accuracy of the SLAM tracking solution in a given environment — and this can vary markedly depending on said environment and the sensing available on the device [13]. Consequently, for large scale co-location in particular, we need the ability to assess the accuracy of any arbitrary combination of MR device and environment, so that we can determine a) whether safe operation is possible, and b) what safety margins need to be incorporated into the experience (e.g., to prevent occluded VR users from accidentally colliding). We could use external optical tracking to provide a ground truth, but this is typically extremely expensive, and indeed undermines one of the key points of using inside-out tracking in the first place: the lack of a reliance on physical infrastructure for tracking. Our challenge is that we can perform a two-point alignment in a given environment, but outside of examining how accurately we can return to these two points after moving through this environment, we cannot assess how well the tracking works throughout the space (e.g., at the edges of a play area). However, if we have two known points in our space, we can determine the position (in x,y) of any other arbitrary point in our play space through trilateration [14].

For this technique, we will need the ability to accurately measure the distance between multiple points in reality. The tool we found best suited to this was commodity laser distance meters, which can be purchased for approximately $30, and typically has millimeter-level accuracy. Let us pick two points at the extremes of our play area, C1 and C2. We will assume these two points lie on our x axis, and the distance between these two points is U, meaning C1 lies at (0,0), and C2 lies at (U,0). For any arbitrary point of interest P, we can determine its (x,y) coordinates by measuring the distance between P and C1 (r_1), and P and C2 (r_2) as follows:

$$x = \frac{r_1^2 - r_2^2 + U^2}{2U} \qquad y = \pm\sqrt{r_1^2 - x^2} \tag{4}$$

Consequently, we can define any number of known points of interest in reality relative to our predetermined alignment points with a high degree of precision using the range finder only. If we then perform a two-point alignment of an MR device to C1 and C2, we can then align the device with any subsequent points of interest and compare the headset's aligned position with the known position in reality, previously determined using the range finder. In this way, we can assess how the accuracy of any given SLAM tracking solution might vary across a given environment.

4.1 Comparing Quest, ARCore, and ZED Mini

To demonstrate this approach, we assessed three SLAM-capable devices: an **Oculus Quest** with a four camera array, a single RGB camera **ARCore**-enabled smartphone (Xiaomi Mi 9T Pro), and a stereo **ZED Mini** RGBD depth camera typically used in robotics and automotive applications, all devices with integrated IMUs that support positional tracking with spatial memory. Given the Covid-19 restrictions at the time of writing, we assessed these devices in a small-scale home play area of $\approx 3 * 2m$ in size, as seen in Figure 7. Our intention was to move the devices around the play area, noting their perceived position in reality at each defined point. We defined 8 points, four covering the extremities of the play area, and four covering the center, as Figure 7 shows. Points (1) and (3) were placed first, and taken as being on our x axis, with (1) being (0,0) and (3)

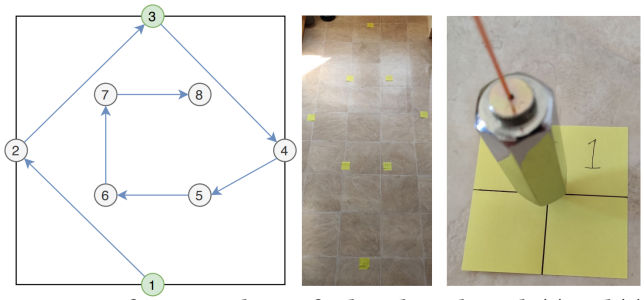

Figure 7: Left: Points chosen for benchmark, with (1) and (3) indicating the alignment points, and the clockwise route indicated (counter-clockwise route was same route in reverse). Middle: Real-world space assessment was performed, with labelled targets. Right: A plumb line being used to line a device up with a target point on the ground.

being measured using a laser distance meter (Lomvum LV-120M, with a range of 120m and a ±0.002m accuracy) to be (0, 2.660). For each of the remaining points, distance measurements were taken from (1) and (3), such that we could determine each point's position in (x,z) on the floor to an estimated accuracy of 0.005m.

Each device was familiarised with the space prior to testing, with the Oculus Quest having a boundary setup, and the ARCore and ZED Mini devices having traversed the environment once fully. A two point alignment with (1) and (3) was then performed for each device, before the devices were taken on 3 laps of our route - going clockwise, and then counter-clockwise along our defined path. At each point on the path, the device was aligned with the target on the floor using a plumb line (a weight on a length of string, attached to the center of each device) to allow for accurate targeting regardless of orientation. The experimenter noted the point id being captured through a dwell interaction (trigger on Oculus Quest, touch screen button on ARCore, keypress on ZED Mini laptop). For the recorded points on each device, we report on the precision (the distance from the mean center of the points) and the accuracy (the distance from the target in reality). It should be noted that this benchmark is not reflective of any one device's performance relative to the others, and no such analysis is performed here. Our intention is to illustrate the degree to which we can assess if a given SLAM device works **in a given environment or context**, and these results are illustrative only of performance in this particular environment.

4.2 Precision and Accuracy Assessment

As can be seen in Table 1 and Figure 8, using trilateration and two point alignment we could assess the robustness of multiple SLAM devices without any external ground truth tracking. We can see that the ZED Mini and Oculus Quest featured near identical performance on average, with a mean localization accuracy of ±0.04m, with a high degree of precision. These results will be influenced by the accuracy of our plumb line targeting, which we would estimate as ±0.01m, but nonetheless such findings would suggest that these devices perform well in the experimenter's kitchen – a suitable environment for a small-scale co-located MR experience. However, the ARCore device featured greater positional deviations, with an accuracy of ±0.18m, and an apparent drift over time (see Figure 8).

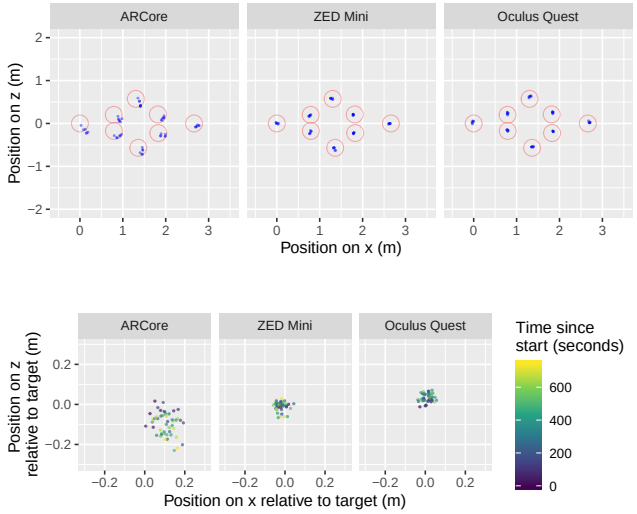

Figure 8: *Top:* Plot of recorded device positions in virtuality (blue points) when aligned with known targets on the ground in reality (red circles). *Bottom:* Plot of device position for every recorded point as an offset of the target point in reality over time.

4.3 Discussion

The ability to assess the accuracy of a SLAM tracking solution is important for any deployment that pushes the boundaries of what these systems are considered capable of. This applies regardless of whether the devices are manually aligned as this paper details, or aligned through synchronised SLAM solutions. Where previous research typically relied on expensive ground truth optical tracking systems, by using trilateration combined with two point alignment we can assess the robustness of any SLAM capable device for any given environment or use case. Armed with such an approach, a practitioner could instrument their given room/arena-scale environment to determine any points where the SLAM tracking struggled, e.g. requiring additional high-contrast features, or explore the upper bounds on the amount of moving proximate persons (e.g. crowds, bystanders) that can be incorporated into the play space without tracking degradation. Our assessment found notable differences in accuracy and precision between a standard ARCore smartphone and the other camera-based SLAM devices. For a practitioner expecting to deploy a combination of these devices co-located into the same environment, they might choose to assess the ARCore device over a greater period of time, to first explore whether the apparent drift it experienced stabilised (which, given the visual landmark recognition of the SLAM tracking, it should). Then, the apparent

Device	Precision (m)		Accuracy (m)	
	Mean	Std.Dev	Mean	Std.Dev
ARCore	±0.10	0.17	±0.18	0.17
ZED Mini	±0.02	0.01	±0.04	0.02
Quest	±0.02	0.01	±0.04	0.02

Table 1: Mean Precision and Accuracy of devices at targets.

inaccuracy of the ARCore devices could be incorporated into the design of the co-located experience: e.g., affording ARCore players a wide virtual berth so that VR users did not accidentally collide with them. In this way, the design can take into account the affordances and capabilities of disparate SLAM devices, validating safe operation and minimizing the possibility of accidental overlaps.

5 GENERAL DISCUSSION

This paper has detailed how the coordinate spaces of multiple disparate SLAM devices can be aligned to enable co-located experiences, formalizing the theoretical accuracy and practical challenges of one- and two-point alignments and providing exemplars of audience, small-scale, and large-scale co-location scenarios. We have also detailed the novel combination of utilising trilateration in conjunction with two-point alignment to assess the robustness of any SLAM device when used in any arbitrary space or context. We can use this approach to evaluate the precision and accuracy of devices without the need for an external ground truth optical tracking, enabling practitioners to validate the suitability of co-located MR deployments in situations that would be considered challenging to visual-inertial SLAM. These challenges might be a result of lighting or weather conditions, moving features such as bystanders or trees in the wind, or novel combinations of MR devices. Regardless, we give practitioners the means to perform these assessments themselves, for their unique use cases.

5.1 Is this really necessary given ____ APIs?

Our focus has been on detailing pragmatic, private, platform-agnostic, easy to implement/understand alignment techniques for MR co-location, formalizing an understanding of how such experiences can be created. In contrast, as previously detailed, there are now a variety of SLAM synchronisation/spatial anchor APIs available from Google, Apple, Microsoft and (soon) Oculus, with varying cross-platform support and reliance on cloud (versus local or P2P) networking for synchronisation. It is therefore pertinent to ask why alignment solutions remain relevant. Eventually, we imagine that there will emerge 'gold standard' cross-platform, standardised, device-agnostic SLAM synchronisation solutions that can operate locally or P2P, able to identify precisely where they are in the world through a sensor-fused combination of SLAM sensing and (when available) GNSS data, an objective that projects like *openarcloud.org* are working toward. However, no such solution currently exists, with the status quo being companies competing for their platforms to become the predominant means by which MR experiences are enacted. Alignment solutions are therefore a useful tool as we journey toward implementations that meet our gold standard requirements. For example, if Oculus release their co-location API and it supports P2P SLAM synchronisation between Oculus devices only, a practitioner could perform an alignment with one Oculus device to then enable ARCore or ARKit devices to be co-located with all Oculus devices in the shared experience. In this way, such techniques can be used not just as the primary means of co-location, but also as a means toward bridging sophisticated platform-specific solutions. And, more broadly, the combination of trilateration and two-point alignment will remain pertinent in assessing the capability of SLAM devices regardless of how co-location is achieved.

5.2 Are there alternative approaches?

It should also be noted that there are other robust approaches that could be considered. For headset-based devices in particular, the same mechanisms by which they track controllers, through embedded IR LEDs with frequency-encoded information, could be used to allow for the tracking of other headsets in a co-located space without necessitating the transfer of point-cloud data or manual alignment - there is no technical impediment preventing a Quest headset from, for example, detecting the position of other controllers in the same play area, and the position of other headsets is known in relation to these controllers. Other signals in the environment could also be used for alignment, much as SynchronizAR [22] used Ultra-Wide Bandwidth distance measurements, for example using triangulation-based localisation techniques to create an alignment, with varying margins for error that might be acceptable for specific use cases. In the end, such solutions may just be stop-gaps, but for specific use cases and user groups these solutions might be more pertinent routes toward immediate ad-hoc co-location.

5.3 Co-located MR and Covid-19

Translational gain is a means by which we can increase the perceived size of a real-world physical space by applying gain to user movement in VR, meaning that, e.g., 1m of movement in reality could equate to 2m or 3m of movement in VR [36]. For those in small scale spaces (e.g., families in small flats), the combination of co-location and translational gain could offer the ability to increase the perceived size of the play area for all taking part. As translational gain is deterministic, it is compatible with the alignment approaches in this paper. We would encourage further research here, particularly in a time when so many find themselves restricted to their homes due to Covid-19 - a co-located MR experience leveraging a combination of VR headsets with translational gain and large immersive environments, and smartphone or headset AR for aligned spectatorship, could create inclusive, shared experiences that defy the perceived limitations of restrictive living spaces that may be becoming all-too-familiar.

6 CONCLUSIONS

Mixed Reality (MR) devices are ever-present in our daily lives; however, building multi-device, multi-user co-located experiences is complex. For practitioners in research and the arts, support is needed for quickly creating low-cost, cross-platform co-located MR experiences. We have provided this support in three ways. Firstly, we have surveyed different approaches toward aligning disparate device coordinate spaces with a real-world environment, formalising the theoretical accuracy and limitations of such approaches. Secondly, we have provided skeleton implementations for audience-based, small-scale and large-scale MR co-location. Thirdly, we have explored how we can assess the suitability of aligned SLAM tracking for any combination of device and environment without the need for an expensive ground truth optical tracking, by using trilateration and a $30 laser distance meter. We open source all our implementations, and hope to democratize the creation of basic co-located cross-platform MR experiences, supporting practitioners and researchers in exploring the shared and social future of MR.

ACKNOWLEDGMENTS

This research received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (#835197, *ViAjeRo*).

REFERENCES

[1] 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

[2] Ryan Anthony J. de Belen, Huyen Nguyen, Daniel Filonik, Dennis Del Favero, and Tomasz Bednarz. 2019. A systematic review of the current state of collaborative mixed reality technologies: 2013–2018. *AIMS Electronics and Electrical Engineering* 3, 2 (2019), 181–223. https://doi.org/10.3934/ElectrEng.2019.2.181

[3] Ramón Argüelles, Matt Wojciakowski, Cory Fowler, and Ross McAllister. 2019. Azure Spatial Anchors overview. https://docs.microsoft.com/en-gb/azure/spatial-anchors/overview

[4] Balthazar Auxietre. 2020. Let's push the boundaries together. https://medium.com/@BalthazarVR/let-us-push-the-boundaries-dee702158919

[5] Paramvir Bahl and Venkata N. Padmanabhan. [n.d.]. RADAR: an in-building RF-based user location and tracking system. In *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, Vol. 2. IEEE, 775–784. https://doi.org/10.1109/INFCOM.2000.832252

[6] Sean Barbeau. 2018. Dual-frequency GNSS on Android devices. https://medium.com/@sjbarbeau/dual-frequency-gnss-on-android-devices-152b8826e1c

[7] Connor DeFanti. 2019. A Quick and Easy Calibration Method. https://frl.nyu.edu/a-quick-and-easy-calibration-method/

[8] Connor DeFanti. 2019. Calibrating the Oculus Quest Guardian. https://frl.nyu.edu/calibrating-the-oculus-quest-guardian/

[9] Connor DeFanti. 2019. *Co-Located Augmented and Virtual Reality Systems*. Ph.D. Dissertation. New York University. https://cs.nyu.edu/media/publications/defanti_connor.pdf

[10] Fredd Duque. 2019. *Six DOF tracking system based on smartphones internal sensors for standalone mobile VR*. Technical Report. http://www.diva-portal.org/smash/record.jsf?pid=diva2:1355314

[11] Hugh Durrant-Whyte and Tim Bailey. 2006. Simultaneous localization and mapping: part I. *IEEE Robotics & Automation Magazine* 13, 2 (jun 2006), 99–110. https://doi.org/10.1109/MRA.2006.1638022

[12] Oculus Facebook. 2020. Horizon. oculus.com/facebookhorizon

[13] Tobias Feigl, Andreas Porada, Steve Steiner, Christoffer Löffler, Christopher Mutschler, and Michael Philippsen. 2020. Localization Limitations of ARCore, ARKit, and Hololens in Dynamic Large-scale Industry Environments. https://doi.org/10.5220/0008989903070318

[14] Michael Geyer and John A. Volpe. 2016. Earth-Referenced Aircraft Navigation and Surveillance Analysis. https://rosap.ntl.bts.gov/view/dot/12301

[15] David Gochfeld, Corinne Brenner, Kris Layng, Sebastian Herscher, Connor DeFanti, Marta Olko, David Shinn, Stephanie Riggs, Clara Fernández-Vara, and Ken Perlin. 2018. Holojam in Wonderland: Immersive Mixed Reality Theater. In *ACM SIGGRAPH 2018 Art Gallery* (Vancouver, British Columbia, Canada) *(SIGGRAPH '18)*. Association for Computing Machinery, New York, NY, USA, 362–367. https://doi.org/10.1145/3202918.3203091

[16] Jan Gugenheimer, Christian Mai, Mark McGill, Julie Williamson, Frank Steinicke, and Ken Perlin. 2019. Challenges Using Head-Mounted Displays in Shared and Social Spaces. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI EA '19)*. Association for Computing Machinery, New York, NY, USA, Article W19, 8 pages. https://doi.org/10.1145/3290607.3299028

[17] Jan Gugenheimer, Evgeny Stemasov, Julian Frommel, and Enrico Rukzio. 2017. ShareVR: Enabling Co-Located Experiences for Virtual Reality between HMD and Non-HMD Users. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*. ACM Press, New York, New York, USA, 4021–4033. https://doi.org/10.1145/3025453.3025683

[18] Ian Hamilton. 2020. Oculus Quest Arena-Scale Projects Persist Despite Unclear Path. https://uploadvr.com/shared-space-facebook-colocation/

[19] David Heaney. 2020. Oculus Quest SDK References Shared-Space Colocation Feature. https://uploadvr.com/oculus-quest-sdk-colocation-api/

[20] Sebastian Herscher, Connor DeFanti, Nicholas Gregory Vitovitch, Corinne Brenner, Haijun Xia, Kris Layng, and Ken Perlin. 2019. CAVRN: An Exploration and Evaluation of a Collective Audience Virtual Reality Nexus Experience. 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, 1137–1150. https://doi.org/10.1145/3332165.3347929

[21] Diana Hu. 2019. The Niantic Real World Platform: Mapping, Sharing and Understanding Reality . https://nianticlabs.com/en/blog/nrwp-update-110619/

[22] 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

[23] Kris Layng, Ken Perlin, Sebastian Herscher, Corinne Brenner, and Thomas Meduri. 2019. Cave: Making Collective Virtual Narrative. In *ACM SIGGRAPH 2019 Art Gallery* (Los Angeles, California) *(SIGGRAPH '19)*. Association for Computing Machinery, New York, NY, USA, Article 1, 8 pages. https://doi.org/10.1145/3306211.3320138

[24] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, Article 36, 17 pages. https://doi.org/10.1145/3173574.3173610

[25] Mirror. 2020. A community replacement for Unity's abandoned UNET Networking System. github.com/vis2k/Mirror

[26] Miscellaneous. 2020. ARWorldMap - ARKit. https://developer.apple.com/documentation/arkit/arworldmap

[27] Miscellaneous. 2020. Share AR Experiences with Cloud Anchors. https://developers.google.com/ar/develop/java/cloud-anchors/overview-android

[28] Mozilla. 2020. Hubs. hubs.mozilla.com/

[29] 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

[30] Spatial. 2020. Collaborate from anywhere in AR and VR. spatial.io

[31] Alex Turner, Matt Zeller, Eliot Cowley, Graham Bury, Jesse McCulloch, Christopher Warrington, and Brandon Bray. 2019. Spatial anchors - Mixed Reality. https://docs.microsoft.com/en-us/windows/mixed-reality/spatial-anchors

[32] Ubisoft. 2016. Star Trek: Bridge Crew. https://www.ubisoft.com/en-gb/game/star-trek/bridge-crew

[33] UploadVR. 2018. Oculus Quest Arena 'Colocation' With Dead And Buried. https://www.youtube.com/watch?v=QJXpHp{_}iQF4

[34] Robby van Delden, Steven Gerritsen, Dirk Heylen, and Dennis Reidsma. 2018. Co-located augmented play-spaces: past, present, and perspectives. *Journal on Multimodal User Interfaces* 12, 3 (sep 2018), 225–255. https://doi.org/10.1007/s12193-018-0269-z

[35] Dominik Van Opdenbosch, Tamay Aykut, Nicolas Alt, and Eckehard Steinbach. 2018. Efficient Map Compression for Collaborative Visual SLAM. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 992–1000. https://doi.org/10.1109/WACV.2018.00114

[36] Graham Wilson, Mark McGill, Matthew Jamieson, Julie R. Williamson, and Stephen A. Brewster. 2018. Object Manipulation in Virtual Reality Under Increasing Levels of Translational Gain. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, Article 99, 13 pages. https://doi.org/10.1145/3173574.3173673

[37] Matthias Wölfel, Daniel Hepperle, Andreas Siess, and Jonas Deuchler. 2020. Staging Location-Based Virtual Reality to Improve Immersive Experiences. *EAI Endorsed Transactions on Creative Technologies* 6, 21 (2020). https://doi.org/10.4108/eai.24-2-2020

[38] Faheem Zafari, Athanasios Gkelias, and Kin K. Leung. 2019. A Survey of Indoor Localization Systems and Technologies. *IEEE Communications Surveys & Tutorials* 21, 3 (2019), 2568–2599. https://doi.org/10.1109/COMST.2019.2911558