Spatial Anchor Based Indoor Asset Tracking

Wennan He
School of Computing
The Australian National University

Mingze Xi*
CSIRO's Data61

Henry Gardner†
School of Computing
The Australian National University

Ben Swift
School of Computing
The Australian National University

Matt Adcock
CSIRO's Data61

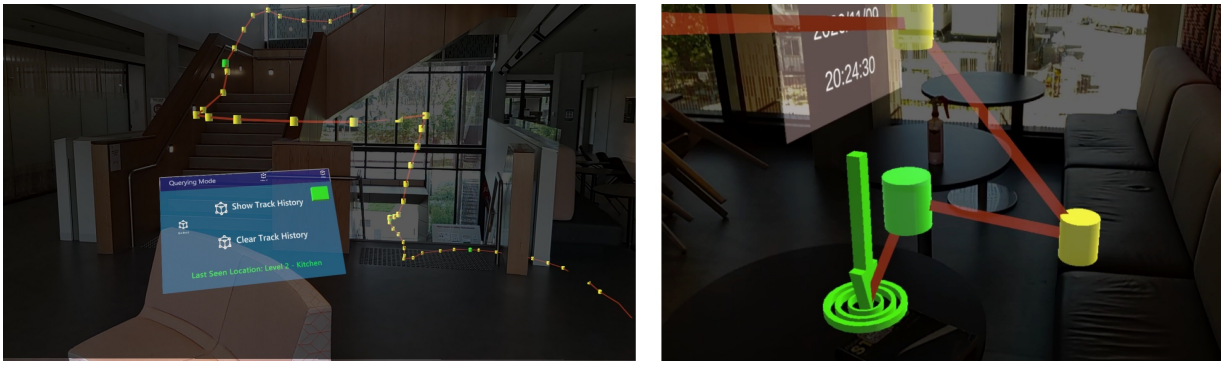

Figure 1: (Left) A tracking scene showing a path of historical target locations. (Right) A tracking scene showing a target asset that has been successfully located. The screenshots have been taken using the AR-IPS demonstrator application described in the paper.

ABSTRACT

Indoor asset tracking is an essential task in many areas of industry such as shipping and warehousing. Widely-used asset tracking technologies typically require supporting infrastructure (signal transponders) to communicate with active tags on the assets. Continuous asset tracking in large indoor spaces like warehouses can be costly, and reliable reference points are needed to calculate asset positions accurately. In this paper, we describe an indoor asset tracking technique for augmented reality (AR) that combines the use of (passive) fiducial markers together with flexible spatial anchors. Our approach, called SABIAT, continuously tracks the approximate location of an asset using spatial anchors and, when needed, the precise location of that asset using fiducial markers. We have applied our SABIAT technique to build a demonstrator system, AR-IPS, to show how assets can be tracked and located inside of a large, multi-level building.

Index Terms: Human-centered computing—Mixed / augmented reality; Information systems—Location based services

1 INTRODUCTION

Indoor positioning (or tracking) systems (IPS) have been a long-term research topic of considerable industry interest. An IPS is a system that continuously tracks the location of an indoor target such as a human or physical asset [9]. IPS systems use a broad range of tracking technologies [9], such as Wi-Fi, RFID, Low-Energy Bluetooth (BLE), Ultra-Wideband (UWB), ultrasound and cameras. All these technologies require specific types of receivers and fixed infrastructure (such as Wi-Fi access points). It can be challenging to build this infrastructure to manage a large number of trackable assets and to modifiy this infrastructure if the tracking requirements change. An additional issue is that IPS systems are often unable to provide accurate 3D spatial locations when tracked objects need to be located very close to one another.

Head-mounted Augmented Reality provides a potential solution to these problems. There are now a number of AR head-mounted displays (HMDs) on the market with different designs and specifications, such as the Microsoft HoloLens 2 [16] and Magic Leap 1 [14]. Previous work has shown that IPS systems using AR devices such as these have the potential to locate and track users and assets in an indoor environment [3, 10]. However, these systems were either not able to track an object on a *continuous* basis [10] or they were difficult to implement without advanced computer vision knowledge [3].

In this paper, we describe a new "**S**patial-**A**nchor-**B**ased **I**ndoor **A**sset **T**racking" (SABIAT) technique, and software plugin, that can be used in AR headsets. SABIAT tracks target objects continuously while they are moved (carried, pulled or pushed) by a user wearing an HMD. SABIAT does not require any additional tracking accessories or hardware apart from the computer vision systems that are typical for HMDs (e.g. a Microsoft HoloLens) and recent mobile devices (e.g. ARKit/ARCore-enabled devices). Using SABIAT we have built a demonstration application that allows assets to be tracked and precisely located inside a large, multi-level building.

The following section reviews related work and provides a motivation for SABIAT. SABIAT itself is described in Section 3 and the demonstrator application, AR-IPS, is described in Section 4. The paper concludes with Section 5.

2 RELATED WORK

2.1 Indoor Positioning Systems

Indoor positing systems (IPS) allow an asset to be localised in an indoor environment via one or more location techniques. Popular technologies include Wi-Fi, Bluetooth, ZigBee, RFID, UWB, visible light, acoustic signal and ultrasound [30]. Using these signals, commonly-used positioning techniques include triangulation, fingerprinting, proximity and vision analysis [9]. These tracking approaches have been proven to work reasonably well within their

*e-mail: mingze.xi@csiro.au
†e-mail: Henry.Gardner@anu.edu.au; Henry Gardner is also a Visiting Scientist at CSIRO's Data61

technological limitations. For example, combining Wi-Fi, IMU and floor information, a smartphone can be used to continuously track a user's indoor position [2] and vision-based localisation systems allow users to locate themselves using captured image frames [6]. There are a large number of commercially-available tracking products, where the tracking targets are often designed in the shape of a small tag or thin card [4, 21]. Detailed surveys of IPS can be found in [5, 9, 27, 30].

2.2 Spatial Anchors

A spatial anchor (or world anchor) is a fixed coordinate system that can be generated by an AR application and tracked by an operating system [18]. Anchors are conventionally used in HoloLens applications to stabilise or persist holograms in the physical space or onto a dedicated surface [12, 13, 19, 24]. Spatial anchors are a critical part of shared AR experiences, where anchors are shared across multiple devices [18, 22]. With recent developments in the ARCore and ARKit software frameworks, anchors are now shareable across different platforms, such as smartphones and tablets. There are also commercially available services, such as Azure Spatial Anchors [15], which provides users with cloud-based anchor management services.

Spatial anchors rely on the accurate localisation of the device, which is primarily achieved through camera-based tracking (with possible assistance from Wi-Fi access points and Bluetooth beacons). For example, the HoloLens 2 implements its visual-inertial SLAM and spatial mapping using data from visible light and depth cameras [23]. Its spatial mapping suffers when the physical environment is poorly illuminated or highly dynamic, such as a space with frequently changing layouts or moving crowds of people. In such situations, the device may not be able to recognise a spatial anchor, causing holograms to "drift" from their original locations.

2.3 Spatial-Anchor-Based Indoor Positioning

In traditional IPS systems that use transponder signals to track assets, one key requirement is the availability of uninterrupted signal broadcasters to act as reference points. Typically, the performance of an IPS can be improved by installing more reference points, although this comes with additional costs.

In AR, spatial anchors can also be used as reference points. When a *user* wearing an HMD moves through space, his or her position can be determined with respect to a nearby spatial anchor. An *asset carried by the user* can also be located accurately and distinguished from other similar assets if the assets have been tagged with fiducial markers. This was demonstrated by He et al. who built an AR application to locate the positions of plant trays in an indoor greenhouse using spatial anchors [10]. Even though their system did not track moving assets continuously, the assets' original and final locations could be determined accurately by observing the fiducial markers on those assets and locating those markers with respect to a nearby spatial anchor.

The two observations that spatial anchors can be used for both (i) the accurate and continuous *tracking of users* and (ii) the *accurate positioning of assets* motivate the SABIAT technique discussed in the next section.

3 SABIAT

In this section, we describe a hybrid technique, and software plugin, for the tracking and positioning of assets in indoor environments that we call SABIAT (for **S**patial **A**nchor-**B**ased **I**ndoor **A**sset **T**racking). SABIAT implements a hybrid tracking approach that combines spatial anchors and fiducial markers to provide both precise and approximate 3D locations of the target object. SABIAT has been written as a plug-in for AR applications to provide location based services.

3.1 Fiducial Markers for Precise Positioning of Assets

Fiducial markers are patterns that can be detected by computer vision algorithms. In an AR application, they allow an accurate calculation of the camera-object pose relative to a marker at a high frame rate, allowing holograms to be registered onto that marker [8,28]. Fiducial markers can be located with high precision in a 3D scene that is understood by the AR system (such as a scene that has been mapped with spatial anchors). Some well-known fiducial marker systems include ARToolKit [11] and ARTag [7,8]. Software libraries such as Vuforia [26] and OpenCV [1] allow any pattern that contains enough feature points to be used as a marker.

From an AR application perspective, a hologram attached to a fiducial marker is represented as a *GameObject* (Unity) or *Actor* (Unreal). An identifiable spatial anchor is also treated as a *GameObject*/*Actor*. This allows the application to continuously calculate and record the relative position between the anchor and a target hologram. When locating a previously tracked asset, the historic position data (and associated anchors) for that object will be retrieved from a server and de-serialised as *GameObjects*/*Actors*. Any holograms can then be positioned accurately (from the user's perspective) relative to the printed markers.

If the location data is stored locally in an AR headset, then users can locate their assets (tagged with fiducial markers) after a restart of the system. Markers can also be located by other AR devices if the location data (including spatial anchors and relative positions) are shared via a server. This approach works with any device that has a spatial mapping capability (e.g. HoloLens, smartphones and tablets with LiDAR cameras), which are necessary for recovering previously generated anchors (see Sect. 2.2). More details on shared AR experiences can be found in [17, 18, 22].

3.2 Head tracking for Approximate Tracking of Assets

In theory, assets that have been tagged with fiducial markers can not only be positioned accurately but they can also be continuously tracked if there is a sufficient number of spatial anchors. However, such a tracking approach requires that a fiducial marker be kept within the view of an AR camera to generate updated locations and orientations. Such a tracking approach is impractical and unsafe. For example, it would mean that users wearing HMDs would not be able take their eyes away from the tracked asset and this could easily cause people to trip over or bump into obstacles. It is therefore necessary to have a supporting mechanism to continuously estimate the movement of an asset even when the marker is out of the user's field-of-view.

Our solution to this problem in SABIAT is to continuously track the accurate location of the AR headset while the wearer is moving the target, and to use this information to update the (approximate) position of the target. Similarly to fiducial markers, the user is represented as a special *GameObject* in Unity called the *Main Camera*. SABIAT calculates and records the user's absolute orientation/direction and relative location to the nearest spatial anchor. The precise location of the user serves as an approximate location of the asset. While the asset is moving, whenever the marker appears in the user's field of view, as sometimes happens when carrying a marked object, SABIAT corrects its estimated position with the precise location of the marker. In SABIAT, the locations of the user and the marker are recorded separately at a high frequency.

SABIAT requires a grid of spatial anchors to accurately calculate the locations of the user and the AR device. According to [18], each anchor can provide coverage for a sphere with a 3-metre radius or a coverage of $28m^2$ mapped onto the floor. The minimal number of anchors can thus be calculated from the floor area. Additional anchors are also required for each partitioned area (e.g. a room). For example, 70 anchors may be sufficient for an empty $2000m^2$ warehouse, while 140 anchors may not be sufficient if the same warehouse is divided into multiple zones by tall shelves. A specialised anchor

placement strategy may be required for complex environments [29].

Our SABIAT software plugin creates a positioning environment for AR applications by combining timestamps, proximity and precise asset locations. Because typical scanning sensors (e.g. depth cameras) are less reliable in outdoor environments, SABIAT has been designed for indoor use at this stage.

4 A Demonstrator Application - AR-IPS

As a demonstration of SABIAT, we have created an application called AR-IPS (Augmented Reality-based Indoor Positioning System). AR-IPS augments SABIAT with a dedicated LAN server, to manage location information and synchronise spatial anchors, and with tools to allow users to locate spatial anchors and to query the paths of tracked assets. Key software development kits and packages used for AR-IPS include Unity, the Mixed Reality Toolkit (MRTK V2.3.0), and Vuforia. AR-IPS has been built for the Microsoft HoloLens (both HoloLens 1 and HoloLens 2). All screenshots of AR-IPS in this paper were captured using a HoloLens 2.

4.1 Implementation

AR-IPS has three modes: tracking, authoring, and querying. On initialisation, the system loads the configuration of previously-placed spatial anchors from a LAN server and enters the tracking mode if anchors are found. If no anchors are found, the system enters authoring mode, which allows an administrator to configure the physical space with spatial anchors. The query mode allows a user to precisely locate a target object by creating a virtual path using its most recent historical locations.

4.1.1 Tracking Mode

As discussed, tracking uses both fiducial markers and spatial anchors. To get accurate locations, AR-IPS uses the Vuforia Engine (v9.28), which recognises and tracks the fiducial markers as *Image Targets* [25]. The markers used in AR-IPS were customised Quick Response (QR) codes. Fig. 2 shows a situation where a fiducial marker is recognised by the Vuforia Engine as having a precise position relative to the "Engineering Mechanics" spatial anchor. In AR-IPS, we used two virtual buttons to manually control the tracking of this specific object. After the user clicks (or speaks) "start", the application starts tracking the location of the asset by sending packets of information ("location records" specifying the asset descriptor and its location) to the server at a fixed update frequency (3Hz for standard operation). The user can then move the asset around until he or she puts it down. The user needs to click the "putting down" button to instruct AR-IPS to record the final location of the asset and to stop tracking.

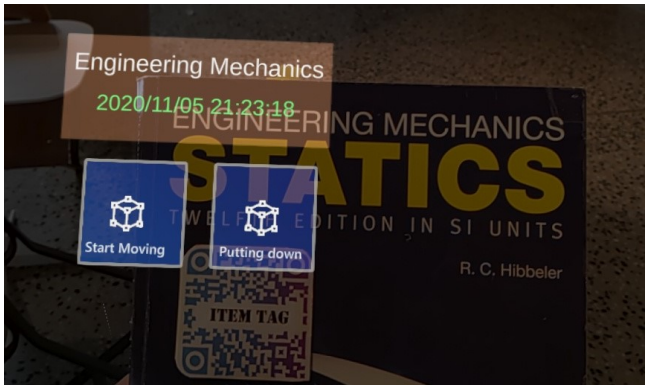

Figure 2: Two buttons and a UI panel are registered on to the fiducial marker when recognised by the Vuforia Engine.

4.1.2 Authoring Mode

Authoring mode allows an admin user to create and interactively-locate spatial anchors in the physical space. A control panel shows the current number of anchors in the space and contains buttons to manage the space (see Fig. 3). The "Create An Anchor" button will instantiate a hologram that appears as an orange cube that can be placed at an appropriate location by the user using far/near interaction (see the MRTK documentation for more details [20]). AR-IPS automatically attaches a world anchor component to that cube, which fixes the anchor to that location in the 3D world as understood by the SLAM algorithm of the HoloLens.

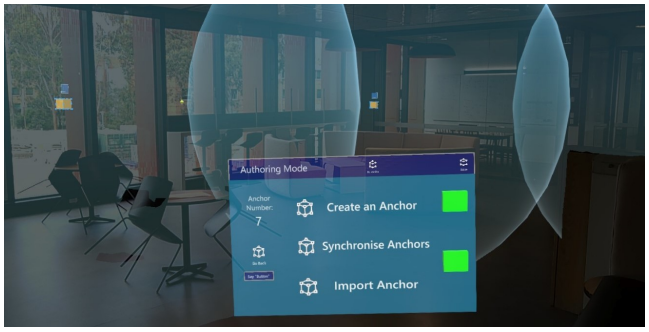

Figure 3: A first-person view of the authoring mode. A authoring panel that contains buttons and two indicators (green squares) allows the user to manage the space. The blue transparent spheres in the background represent the coverage of a spatial anchor.

To prevent redundant anchors from being generated, an anchor can only be placed if no other anchor is within a 3m radius of the user. This is indicated by a green indicator next to the "Create An Anchor" button. Once an anchor is placed in the space, its approximate coverage (a sphere of radius 3m) is visualised to help the user check whether the spatial anchors fully cover the indoor space. Each anchor has a unique ID and an optional location name.

To finish authoring the space, the user needs to synchronise the local anchor data with the anchor server using the "Synchronise Anchors" button. AR-IPS will export and upload all spatial anchors in a batch transaction. An indicator next to this option changes from yellow to green (not shown in the figure) if the synchronisation process is successful, otherwise it shows as red indicating a failure.

4.1.3 Querying Mode

AR-IPS retrieves location records of tracked assets from a (MySQL) database on the server. These records are encoded in JSON format. In Unity, *JsonUtility* was used to de-serialise location records and to then spawn a set of *GameObjects* corresponding to the recorded locations. AR-IPS comes with an optional algorithm to pre-processes the location records so that they appear as a continuous path from the initial to the final asset locations.

4.1.4 Local Anchor Server

AR-IPS uses a dedicated LAN "anchor server" with a custom Python app running on CentOS7 to manage the large amount of spatial anchor data generated by the system. Spatial anchors and location records are handled by Python scripts deployed on the LAN server. Anchor data is stored as a binary file, while location records (JSON payloads) are stored in a MySQL database. The anchor server uses HTTP to handle the transfer of spatial anchor data, and raw TCP sockets to handle location records. Multiple HoloLenses can be connected to the LAN server via a consumer-level wireless router.

4.2 Demonstration

To verify the capabilities of AR-IPS to track and locate a physical asset, we conducted a series of trials inside a large 3-level building. An anchor server (Sect. 4.1.4) was set up on a laptop and connected to a private network. A Microsoft HoloLens 2 with pre-installed AR-IPS was also connected to the same private network. We then configured the building using the authoring tool (Sect. 4.1.2), which generated and synchronised 43 spatial anchors. In the tracking mode (Sect. 4.1.1), one user tracked multiple objects with various movement paths. After this, a different user was asked locate the object using the querying tool (Sect. 4.1.3).

One of these trials is shown in Fig. 1. The path on the left hand side of Fig. 1 shows the trace of the target object—in this case, a book. The smaller yellow cylinders represent the approximate locations determined by the movement of the headset; the larger green cylinders verify that the asset has been observed and thus confirm its precise location. Note that these green cylinders have been moved so that they form a smooth path with the adjoining orange cylinders. Even though their positioning is not precise, they do contain the precise location of the asset which is available for download by querying a green cylinder. Each cylinder also carries tracking metadata (e.g. a timestamp), which will be available when examined by the user (using eye-tracking on a HoloLens 2). A smooth path connecting all the cylinders leads the user to the last-seen location of the target object (pointed to by a green arrow on the right hand side of Fig. 1). Logs of the trials showed that the location of the user (and assets) were tracked continuously throughout this 3-level indoor environment.

4.3 System Latencies

Without a careful implementation of AR-IPS the latency in accessing the spacial anchor and asset path data can be significant, leading to a frustrating user experience.

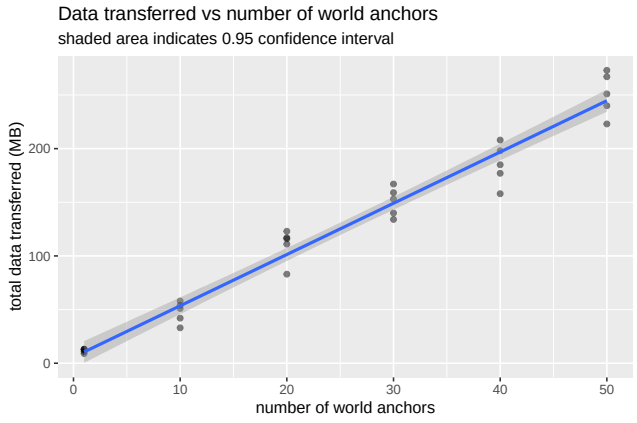

Figure 4: Measured relationship between the number of spatial anchors and the amount of data transmitted across the network.

A possible way to implement an application like AR-IPS would be to use a commercial system such as Microsoft Azure Spatial Anchors (ASA). Indeed, ASA was initially used for AR-IPS, but we found that it had an unacceptable latency when synchronising spatial anchors (possibly due the slow connection speeds to the remote servers from the part of the world in which this work was conducted). This is shown in Fig. 4 which shows the results of a set of performance trials inside of a 3-level office building. In these experiments, one of the authors first scanned the building wearing a HoloLens 2, and a set of anchors (corresponding to configurations of 1, 10, 20, 30, 40 and 50 anchors) was randomly placed in authoring mode. The quantity of transmitted anchor data was recorded and all anchors were then cleared for the next round (with 5 rounds for each configuration). The results in Fig. 4 show a clear linear relationship between the quantity of data transferred and the number of anchors. For the largest configuration, AR-IPS generated approximately 251MB of data for 50 anchors ($n = 5, \sigma = 20.3$). The corresponding latencies are considerable: We found that initialising a system of 50 anchors took 200-250 seconds using ASA over repeated evaluations. By contrast, this initial download took only 15-30 seconds using our dedicated server.

In the querying mode, time cost of de-serialising and rendering the tracked asset paths (querying time) mean that there is a trade-off between the querying time and the distance between the rendered path cylinders (the inter-cylinder distance). Fig. 5 shows some measurements of the relationship between the querying time and the inter-cylinder distance. Low inter-cylinder distances give rise to paths that appear very smooth in space, but have a longer querying time due to the significantly increased number of cylinders. In contrast, high inter-cylinder distances make the paths appear too shattered to follow. We found that AR-IPS offers an acceptable querying time ($n = 5, \mu = 2.8, \sigma = 0.5$), a smooth path, and a stable frame rate of 60 FPS [19], when the inter-cylinder distance is set to be 0.5m. As a result, a default tracking frequency of 3Hz was chosen for AR-IPS to ensure sufficient tracking records (cylinders) with an assumption that users have a constant walking speed of 1.4m/s. The generated paths shown in Fig. 1 correspond to this frequency.

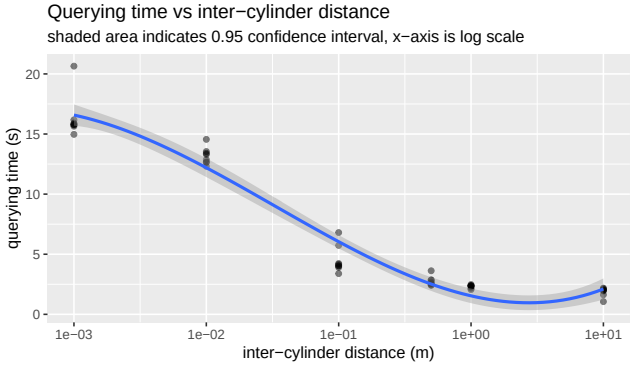

Figure 5: Measured querying time as a function of the inter-cylinder distance.

5 Conclusion and Future Work

This paper has described a hybrid asset tracking/positioning technique and software plugin, SABIAT, for use by augmented reality applications. A demonstrator application, AR-IPS, showed that SABIAT can be implemented to run efficiently using a HoloLens 2 in large spaces using a large number of spatial anchors provided that the spatial anchors are managed using a dedicated LAN server. The use of a large grid of spatial anchors has been possible even though this is discouraged by the official Microsoft documentation [18]. Our SABIAT plugin can be used for a range of applications and it can be bundled in with the authoring and querying tools that we have demonstrated in AR-IPS.

One avenue for future work is improving the robustness of our approach with algorithms to recover lost anchors when inaccurate or noisy spatial locations (e.g. due to poor lighting conditions) are encountered. We are also interested in tracking precise asset locations without having markers, perhaps using a lightweight deep-learning model that can be trained within an AR application to detect target objects in real-time. Including automatic anchor generation in our authoring mode is another interesting avenue of work to explore in the future.

REFERENCES

[1] G. Bradski. The OpenCV library. *Dr. Dobb's Journal of Software Tools*, 2000.

[2] J. L. Carrera V., Z. Zhao, T. Braun, Z. Li, and A. Neto. A real-time robust indoor tracking system in smartphones. *Computer Communications*, 117:104 – 115, 2018. doi: 10.1016/j.comcom.2017.09.004

[3] L. Chen, Y. Zou, Y. Chang, J. Liu, B. Lin, and Z. Zhu. Multi-level scene modeling and matching for smartphone-based indoor localization. In *2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pp. 311–316, 2019. doi: 10.1109/ISMAR-Adjunct.2019.00-22

[4] CSIRO. BLEAT. Accessed on: Nov. 3, 2020. [Online]. Available: https://products.csiro.au/bleat/about-bleat/, 2020.

[5] P. Davidson and R. Piché. A survey of selected indoor positioning methods for smartphones. *IEEE Communications Surveys Tutorials*, 19(2):1347–1370, 2017. doi: 10.1109/COMST.2016.2637663

[6] J. Dong, M. Noreikis, Y. Xiao, and A. Ylä-Jääski. ViNav: A vision-based indoor navigation system for smartphones. *IEEE Transactions on Mobile Computing*, 18(6):1461–1475, 2019. doi: 10.1109/TMC.2018.2857772

[7] M. Fiala. ARTag, an improved marker system based on artoolkit. *National Research Council Canada, Publication Number: NRC*, 47419:2004, 2004.

[8] M. Fiala. ARTag, a fiducial marker system using digital techniques. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 2, pp. 590–596 vol. 2, 2005. doi: 10.1109/CVPR.2005.74

[9] Y. Gu, A. Lo, and I. Niemegeers. A survey of indoor positioning systems for wireless personal networks. *IEEE Communications Surveys Tutorials*, 11(1):13–32, 2009. doi: 10.1109/SURV.2009.090103

[10] W. He, B. Swift, H. Gardner, M. Xi, and M. Adcock. Reducing latency in a collaborative augmented reality service. In *The 17th International Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI '19. ACM, 2019. doi: 10.1145/3359997.3365699

[11] I. P. H. Kato, M. Billinghurst, and I. Poupyrev. ARToolkit user manual, version 2.33. *Human Interface Technology Lab, University of Washington*, 2, 2000.

[12] L. Kästner and J. Lambrecht. Augmented-reality-based visualization of navigation data of mobile robots on the Microsoft Hololens - possibilities and limitations. In *2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pp. 344–349, 2019. doi: 10.1109/CIS-RAM47153.2019.9095836

[13] A. Luxenburger, J. Mohr, T. Spieldenner, D. Merkel, F. Espinosa, T. Schwartz, F. Reinicke, J. Ahlers, and M. Stoyke. Augmented reality for human-robot cooperation in aircraft assembly. In *2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, pp. 263–2633, 2019. doi: 10.1109/AIVR46125.2019.00061

[14] Magic Leap. Magic Leap 1. Accessed on: Nov. 3, 2020. [Online]. Available: https://www.magicleap.com/en-us/magic-leap-1, 2018.

[15] Microsoft Corporation. Azure Spatial Anchors overview. Accessed on: Jan. 21, 2020. [Online]. Available: https://docs.microsoft.com/en-us/azure/spatial-anchors/overview, 2019.

[16] Microsoft Corporation. Microsoft HoloLens. Accessed on: Nov. 3, 2020. [Online]. Available: https://www.microsoft.com/en-us/hololens, 2019.

[17] Microsoft Corporation. Shared experiences in mixed reality. Accessed on: Nov. 3, 2020. [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/platform-capabilities-and-apis/shared-experiences-in-mixed-reality, 2019.

[18] Microsoft Corporation. Spatial Anchors. Accessed on: Nov. 3, 2020. [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/design/spatial-anchors., 2019.

[19] Microsoft Corporation. Hologram stability. Accessed on: Jan. 21, 2020. [Online]. Available:https://docs.microsoft.com/en-us/windows/mixed-reality/develop/platform-capabilities-and-apis/hologram-stability, 2020.

[20] Microsoft Corporation. How to add near interaction in MRTK. Accessed on: Jan. 21, 2020. [Online]. Available: https://microsoft.github.io/MixedRealityToolkit-Unity/Documentation/Input/HowToAddNearInteractivity.html, 2020.

[21] NutTAG Australia. NutTAG Focus. Accessed on: Nov. 3, 2020. [Online]. Available: https://nuttag.com.au/collections/nuttag-focus, 2020.

[22] I. Sluganovic, M. Serbec, A. Derek, and I. Martinovic. HoloPair: Securing shared augmented reality using microsoft hololens. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, ACSAC 2017, p. 250–261. ACM, 2017. doi: 10.1145/3134600.3134625

[23] D. Ungureanu, F. Bogo, S. Galliani, P. Sama, X. Duan, C. Meekhof, J. Stühmer, T. J. Cashman, B. Tekin, J. L. Schönberger, et al. HoloLens 2 research mode as a tool for computer vision research. *arXiv:2008.11239*, 2020.

[24] R. Vassallo, A. Rankin, E. C. S. Chen, and T. M. Peters. Hologram stability evaluation for Microsoft HoloLens. In M. A. Kupinski and R. M. Nishikawa, eds., *Medical Imaging 2017: Image Perception, Observer Performance, and Technology Assessment*, vol. 10136, pp. 295 – 300. International Society for Optics and Photonics, SPIE, 2017. doi: 10.1117/12.2255831

[25] Vuforia. Vuforia developer library — image targets. accessed on: Nov. 3, 2020. [online]. available: https://library.vuforia.com/features/images/image-targets.html, 2019.

[26] Vuforia. Vuforia Engine. Accessed on: Nov. 3, 2020. [Online]. Available: https://www.ptc.com/en/products/vuforia/vuforia-engine, 2020.

[27] Z. Wang, Z. Yang, and T. Dong. A review of wearable technologies for elderly care that can accurately track indoor position, recognize physical activities and monitor vital signs in real time. *Sensors*, 17(2):341, 2017. doi: 10.3390/s17020341

[28] Xiang Zhang, S. Fronz, and N. Navab. Visual marker detection and decoding in ar systems: a comparative study. In *Proceedings. International Symposium on Mixed and Augmented Reality*, pp. 97–106, 2002. doi: 10.1109/ISMAR.2002.1115078

[29] H. Xing, C. Yang, X. Bao, S. Li, W. Gai, M. Qi, J. Liu, Y. Shi, G. De Melo, F. Zhang, and X. Meng. MR environments constructed for a large indoor physical space. In *Advances in Computer Graphics*, pp. 132–144. Springer, 2020. doi: 10.1007/978-3-030-61864-3_12

[30] F. Zafari, A. Gkelias, and K. K. Leung. A survey of indoor localization systems and technologies. *IEEE Communications Surveys Tutorials*, 21(3):2568–2599, 2019. doi: 10.1109/COMST.2019.2911558