Look Behind You! – Using a Face Camera for Mobile Augmented Reality Odometry

Jan Čejka[1] and Fotis Liarokapis[2]

[1] Faculty of Informatics, Masaryk University, Brno, Czech Republic, xcejka2@fi.muni.cz
[2] Research Centre on Interactive Media, Smart Systems and Emerging Technologies (RISE), Nicosia 1011, Cyprus, f.liarokapis@rise.org.cy

Abstract. Augmented reality applications provide new ways of presenting cultural heritage assets thanks to the recent advancement in the field of smart devices. Unfortunately, the construction of the hardware and lower computational power of mobile processors limit the potential of these applications. Namely, almost all current visual-inertial odometry libraries employed in smartphones require the real tracked objects to be close and contain distinguishable features, which is an issue when observing large virtual structures outdoors like historical buildings or objects on plain walls of halls or museums. This paper exploits the possibility of using the face cameras available in mobile devices for augmented reality tracking. It designs a prototype composed of iPhone and iPad devices and evaluates its contribution in two scenarios that current systems cannot handle. The results reveal the clear benefit of this approach for the cultural heritage, allowing it to operate in situations when users look up in the sky to see the roof of virtual buildings, or when they move closer to a white wall to perceive details of a virtual painting. In the end, the paper discusses the system's limitations and proposes solutions to them.

1 Introduction

Augmented reality (AR) is on its way to become a part of our everyday lives. It has the capability not only to enhance the existing objects and visualize the future, but it also allows us to observe lost artefacts and understand our history. Increased computational power of mobile devices allows them to process input images in real time, and with the aid of inertial measurement units (IMU), smartphones are able to precisely track users position, which is essential for AR applications. Operating systems now include libraries that simplify development of such applications with features like estimation of the reflection map and of environmental lighting, recognition of basic objects with artificial intelligence (AI), cooperation between users, and other [1,11].

Despite the vast capabilities, limited resources of these devices constrain the algorithms to utilize only the data of the rear camera, IMUs, and ambient light sensors. This restricts the applications, because they must force users to aim the device at nearby distinguishable objects, usually at tables or down at the ground.

When the user rotates its view and the device cannot see any of such objects, the tracking stops working and the AR experience is weakened. This can be partially compensated with other localization technologies like global navigation satelite systems (GNSS), but their precision is not sufficient to closely examine virtual objects. For these reasons, it is still very hard to provide AR tools for observing virtual objects on clear walls like virtual paintings in digital museums, or to create electronic guides displaying non-existing cultural heritage structures over their remains at archaeological sites.

Mobile devices are equipped with a front camera (face camera) and a set of one or more rear cameras, but hardware limitations disallowed developers from using the face camera and any of the rear cameras at the same time. Fortunately, recent developments indicate that this is no longer an issue, and the latest hardware is capable of processing both image streams from the face and the rear camera at the same time, which can be used in video conversations [2]. However, this also opens new options for localizing the device in the environment.

This paper enhances the abilities of AR at cultural heritage sites and investigates the possibility of using the face camera to maintain the tracking when the data from the rear camera is insufficient. It presents a system that tracks the device position by processing the visual information in front of the device as well as behind it. This system is tested with a prototype consisting of two joined mobile devices oriented in opposite directions, and evaluated in two experiments, one with participants observing a non-existing historical church, and one with them examining a virtual painting on a clear white wall. To our knowledge, this is the first tracking system designed for mobile AR applications that utilizes the data from behind the user to improve tracking in situations when the rear camera is not sufficient.

The main contributions of this paper are:

- a design of a system that builds the tracking on a combination of the data in front of the device with the data behind the device;
- an evaluation of this system in two experiments: an outdoor experiment with users walking around a non-existing historical building, and an indoor experiment with users observing a virtual painting on a clear white wall.

2 Related Work

Odometry algorithms solve the problem of localization, which includes feature detection and matching for the purpose of finding the viewer in a known area and updating its location. This is often combined with the problem of mapping, which is building a map of the surrounding area that is used for the localization. Algorithms that incorporates and optimizes both parts together are denoted as simultaneous localization and mapping (SLAM).

Many algorithms for single-camera visual-only SLAM are based on the work of Klein et al. [15] presenting an algorithm that separates the tracking and mapping into two parallel components that are updated at different rates. Various natural features are exploit for fast detection and matching, like ORB features [20] or line

features [26,10,37]. Some authors decide not to use features and perform a direct evaluation of intensity changes between subsequent frames of the video [6,29,7,8]. Benefits of stereo cameras were explored in works of Park et al. [24] or Hsieh et al. [13]. SLAM algorithms based only on the visual data are prone to blurred images, caused especially by large rotational movements of the cameras. This can be compensated very well by incorporating data from inertial sensors, combined with feature trackers [19,21,17,27] or direct trackers [3,4], and optimized for mobile devices [18]. Visual-inertial odometry (VIO) SLAM systems with a pair of stereo cameras were exploit in the work of Leutenegger et al. [17] or Usenko et al. [32]. Non-stereo multiview VIO SLAM systems were exploit mostly in the field of robotics and used various distribution of cameras, e.g., multiple pairs of stereo cameras [23,12], cameras directed forward and down [36,5], and cameras with non-overlapping views [31]. Karrer et al. presented a system for collaborative AR [14], which combined maps of several mobile AR systems into a large shared map. All of these systems were designed for the general problem of mapping, however, and none of them were tested in situations when one of the cameras could not track its surrounding.

There are many commercial libraries for indoor AR applications developed for common mobile devices like smartphones or tables [35,25], and some of them are also integrated directly into the mobile operating systems [1,11]. They support markerless VIO SLAM, estimation of the environment lighting conditions, detection of vertical and horizontal surfaces, and recognition of predefined 2D images. They can also share the tracking data between multiple devices to create a collaborative AR experience. There are also open-source libraries for building AR applications [28]. Outdoor AR systems are often related to applications for sites with cultural heritage importance [34]. Galatis et al. [9] described KnossosAR, a system designed for ancient sites that handled occlusions of real objects on the virtual scenery. Some mobile AR systems were also aimed at underwater environments [22] and underwater cultural heritage sites [33]. Other systems [16,30] showed historical buildings in existing cities, but none of them tackled the problem of insufficient amount of features to track.

3 System Description

Our system is designed for Apple devices and bases its tracking on the ARKit library, which is available for the general public. Unfortunately, this library can use only the rear camera to track the environment, using the face camera for tracking in AR applications is not supported. For this reason, our system consists of two mobile devices, the iPad device (iPad Pro, 2nd generation, 10.5-inch, iPadOS 13.4), which tracks the world in front of the user, provides additional tracking data with its IMU, displays the augmented reality content, and receives the user input, and the iPhone device (iPhone XS, iOS 13.4), which uses its rear camera to track the scenery behind the user (see Figure 1).

The iPhone is placed in the lower left corner on the iPad with its screen facing the screen of the iPad. The iPhone is secured with a wedge, pressed to the

Fig. 1: Left: The tracking system consists of two components: an iPad (receives the data from its rear camera, its IMU, and the user, and displays the AR content), and an iPhone (receives the data from its rear camera). Right: The iPhone being placed on the iPad device with its rear camera directed behind the user, and together, the devices work as one stable unit.

iPad with rubber bands, and separated with a soft pad to protect the screens. Despite the simplicity, this setup was found to be compact, stable, firm, and easy to manipulate, and the iPhone's camera is very close to the actual position of the iPad's face camera. Relative position between the devices was measured with a precision lower than one centimeter, and their relative orientation was estimated from the setup.

The devices communicate via Apple's Multipeer Connectivity framework. The iPhone tracker sends updates in measured position and orientation, status of tracking, number of tracked features, and a timestamp of each data. In the opposite direction, the iPad tracker sends commands to reset the tracking at the start of each user test. For testing purposes, the iPhone also sent the image stream from its rear camera in a low quality, which was then presented in the lower right corner of the iPad.

3.1 Tracking

The system is composed of three tracking units: an iPad tracker, an iPhone tracker, and a Core Motion tracker. The first tracking unit, the iPad tracker, is realized with the ARKit library and tracks the objects located in front of the user. The iPhone tracker is similar and tracks the objects behind the user. The last tracker, the Core Motion tracker, is based on Apple's Core Motion framework and uses the inertial sensors to track the orientation of the iPad.

The tracker state is derived from four states of ARKit mapping. When the ARKit library reports that it is in a *mapped* environment or *extending* its internal map, the tracking provides good estimates of device location. The *limited* state indicates that the system can provide some estimates of the position, but the area is not sufficiently mapped. If the mapping has just started or is not consistent, the

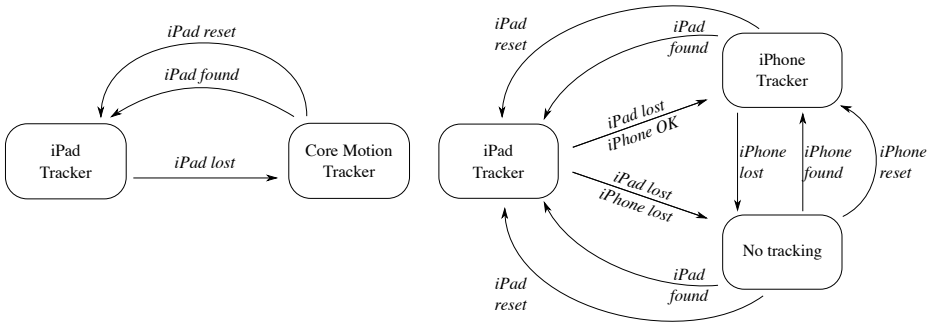

Fig. 2: Left: The orientation is tracked by switching between iPad tracker and Core Motion tracker. Right: The position is tracked by switching between iPad tracker and iPhone tracker, when they are capable of tracking.

tracking is *not available*. Our system accepts the *mapped* and *extending* tracking, and includes also the *limited* tracking if the number of features is at least 100. In other cases, it still uses the position obtained from ARKit for one second, since the preliminary tests found that the error in position is tolerable and the library often relocates itself. After this second, the tracker is considered lost.

The system switches between the tracking units according to their ability to track their location, see Figure 2. Computing the orientation is easier; the system uses ARKit and iPad data, and if the library cannot track the location, it switches to the Core Motion tracker (denoted as *iPad lost*). When the iPad tracker relocates itself (*iPad found*), the system drops the orientation from the Core Motion tracker and continues using the iPad tracker. When ARKit decides to reset the tracking session (*iPad reset*), the system aligns the last known orientation with the new ARKit session and continues tracking.

The position is computed differently. The system again uses the position from the iPad tracker, and when it becomes lost, it starts using the iPhone tracker (*iPad lost, iPhone OK*). If the iPhone tracker cannot measure the position, the system stops updating the position and updates only its orientation (*iPad lost, iPhone lost*). This also happens when the iPhone stops tracking the position (*iPhone lost*). When the iPhone tracker successfully relocates itself (*iPhone found*), the system continues with updating the position. Similarly, if the iPhone tracker resets the tracking session (*iPhone reset*), the system aligns the new session and continues tracking. When the iPad tracker successfully locates itself in its environment (*iPad found*), the system drops the position updated by iPhone and uses the iPad position. If the iPad tracker is reset (*iPad reset*), the system aligns the new session and continues tracking.

4 Experiment

Our solution was evaluated in two experiments. The first experiment focused on users observing large objects in outdoor environments, as illustrated in Figure 3.

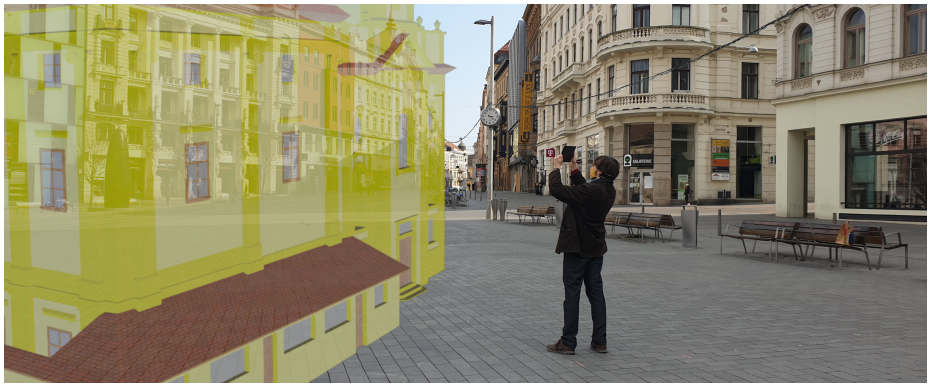

Fig. 3: A participant observing a historical church

A virtual model of a historical church was placed in the city center where it stood before it was demolished. The model of was unlit, textured, and consisted of approximately 19000 triangles. It spanned the area of 32×30 meters, and was 26 meters high. The shape of the city square is approximately a triangle of length $130 \times 150 \times 170$ meters. In this experiment, five users (three males and two females; one participant in age category 18–25, two participants of age 26–33, and two participants more 50 years old) walked around the church and observed how it fit the square. They were not limited by time and were instructed to look up to see its upper floors and its roof. The application logged the tracking status, the orientation, and the time at which the frame was recorded.

The second experiment focused on people observing objects on surfaces with no features to track, like white walls. The users were asked to see a virtual painting on a real wall and search the painting for specific small details. The virtual model of the painting was realized with a simple textured rectangle of size 1.35×0.9 meters and located in a room of size of 4.2×5.0 meters. This test was completed by four users (two males and two females; two participants of age 26-33, and two participants more than 50 years old). Again, they were not limited by time or distance at which they should search the painting. As they moved closer, the rear camera it saw only the wall and could not track the user's position. The application logged similar data as in the first experiment, but in this test, the device recorded the distance from the virtual object instead of its orientation.

In the beginning of both experiments, the participants were familiarized with the procedure, and they were instructed to tilt the device to avoid being in the field of view of the face (iPhone) camera. The supervisor then initialized the system and placed the virtual object at its correct location. The participants attended a single session of each experiment and spent from 3 to 6 minutes with the first part and from 1 to 2 minutes with the second part.

5 Results

Evaluation of the experiments was based on the comparison of the number of frames in which our system and sole face and rear trackers provide position data. The results include the one-second interval of accepting the estimated location in insufficient conditions as described in Section 3.1, which is sufficient for presenting the AR content.

Figure 4 presents the results of the first test evaluating the system in the outdoor environment. It shows that when the pitch is negative (the users look down), the rear camera tracker (iPad) is able to track the user without any significant issues. The face camera tracker (iPhone) cannot accurately follow the user and tracks the user in rare parts when there is a static object to track, like a lamp. The users spent most of their time looking ahead or slightly down (in range from -20 to 10 degrees), and in these situations, the combined system provided the tracking sufficient for AR applications in more than 50 % of frames. It should be noted that the graphs show only the results of position tracking; the orientation is always available, thanks to the Core Motion tracker. When the pitch is around zero, the view of users is directed ahead, and both trackers have the ground in their field of view. Thanks to this, the number of tracked frames increases for the front-camera tracker. The performance of the rear camera tracker is lower, since the frames contain fewer close features to follow than when the view of users is directed down.

When the pitch is positive, users look up and the usability of the rear camera tracker is very low. The system compensates this with the data of the face camera tracker, which is now directed down. However, the performance of the face tracker is still very low when compared to the rear tracker. The reason is that the users were fully immersed in the experience and forgot to tilt the device to be out of the iPhone's field of view. The presence of both moving and static objects in the camera stream confused the tracker, which could not provide any data. The figure also shows an unexpected peek in performance at the beginning of the

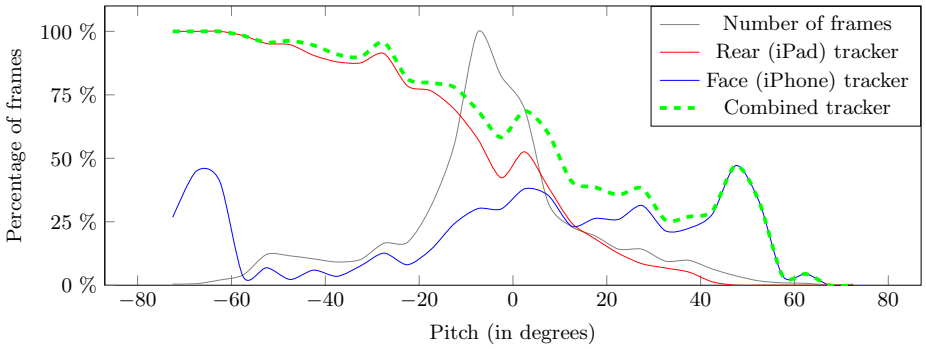

Fig. 4: The performance of the tracking solutions when users observed the virtual model of the church

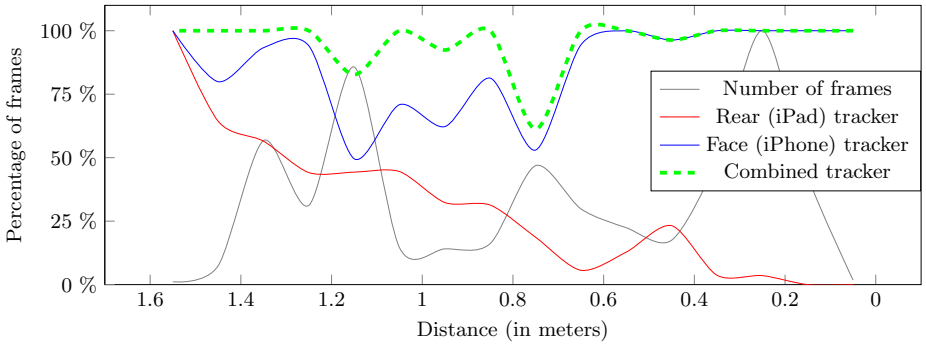

Fig. 5: The performance of the tracking solutions when users observed the virtual painting on a clear wall

graph when the pitch is negative and the device is directed to the ground. We assumed that this represents some very rare occasions when users asked for a help during the testing, stopped walking, and thus they became the static object that the iPhone camera required for tracking.

Figure 5 shows the results of the experiment with users observing details of a virtual painting on a real clear wall. Performance of the rear-camera (iPad) tracker was decreasing as the users moved closer to the wall, since the camera started losing recognisable object to follow. The face-camera (iPhone) tracker had no problem with tracking, as it continued to track the room behind the user. In this experiment, the participants focused more on being out of the field of the view of the iPhone camera than in the previous experiment, which allowed the tracker to operate in most of the frames. The figure confirms that our system utilized the data from both trackers and provided the tracking in more than 50 % of frames, and more than 75 % of all frames except a peak at around 0.7 meters. The users spent most of their time at a distance between 1.1 and 1.4 meters, at which they started the experiment, and at the distance of around 0.2 meters, at which they searched the painting for the objects.

The participants enjoyed the first experiment very much, especially the possibility of seeing the non-existing historical church at is former place. They mentioned that in some situations, they had problems with walking around the church as it was moving with them – this happened when the system lost the tracking from both cameras and could use only the data from the inertial unit. Regarding the second experiment, the participants reported that the resolution of the painting was very low, they saw individual pixels of the image and had difficulties to recognize the object they were supposed to find. Despite this, none of them reported the impossibility of moving closer to the painting, which indicates that the system had no problems with tracking the user's position.

5.1 Discussion

The ARKit library assumes that the tracked environments is static and objects do not move. This was found to be a major issue of the face-camera (iPhone) tracker, since the participants were in the field of view of this camera, which confused the tracker. Although the users were asked to slightly rotate the device during the experiments, this was more successful in the second test. Additionally, the preliminary tests found that the automatic focus caused blurring of the background and a loss of tracking when the users accidentally got into the field of view, so it was disabled for the iPhone and fixed at 2 meters (however, the automatic focus was enabled on iPad). Another possibility is to detect the person in the image, mask it out, and use only the background for tracking, but this is not supported by ARKit, which does not allow the developers to change the camera image before it is processed.

When the individual trackers got lost, it took a few seconds to create a new map of the area and restart tracking. This time could be decreased if the trackers shared the map of the surroundings. Although ARKit supports sharing the maps with *collaborative sessions*, the preliminary tests found that the relative position between the devices was erroneous, and sharing a map of the environment did not decrease the time to initialize the tracking, so this feature was removed from the final version of the system.

The experiments did not evaluate the precision of the tracking, which we expect to be comparable to the precision of individual trackers, because the system has no effect on the mapping capabilities of individual trackers. The precision of the whole system is affected by inaccuracies in the measured relative position and orientation between the devices and will influence the final position if the distance travelled without tracking is large, but this was not an issue. Our system accumulated large errors in position only in situations when it lost the tracking from both devices, which occurred only during the outdoor experiment when one device lost the tracking and the second device was building its map of the environment. It was observed by just a few users only for brief moments, and since the evaluation of this experiment was not based on the position of the user, this problem was ignored. Such error in position can be reduced by adding additional points of reference into the environment (like markers) or by incorporating GNSS. Errors in the orientation were vary rare and happened at the beginning. In such situations, the test was restarted and the measured data was ignored. These problems could be avoided by incorporating the magnetometer, but preliminary experiments showed that it can easily start reporting invalid data due to the presence of artificial objects that deform the magnetic field of the Earth, so it was not integrated into the system.

The system requires the area behind the user to contain features to track when observing virtual paintings on clear walls, but this can be an issue in two scenarios. The problem appears in corridors when the user is surrounded by clear walls, since the system cannot find any distinguishable features in front of the users as well as behind them. It is also observed in large halls, since the objects

behind the user are not sufficiently close. These limitations can be overcome only by incorporating GNSS.

Modern smart devices are equipped with time-of-flight sensors, which allow them to measure the distance of close objects. Such devices were not tested in our experiments, but despite the fact that they can obtain the distance to clear walls, we assume they will have problems with tracking changes in the user's position when moving along the wall, because the grid that is projected and recorded by such sensors travels as well. Also, they will not help outdoors, since there are no close objects when users look up. Despite this, these sensors can still help with tracking, as they are able to decrease the time required to initialize the trackers.

6 Conclusion

This paper focused on enhancing the capabilities of AR in digital museums and cultural heritage sites. It presented a system for mobile AR applications that tracked the position of users also in situations with no recognizable objects in front of them. Unlike the state-of-the-art systems that use only the rear camera of the device, our system added the face camera oriented behind the user to provide the data when the information obtained from the rear camera was insufficient. A prototype of this system consisted with two devices, an iPhone and iPad, and was tested in two use cases – one with participants observing a virtual model of a historical church at its formal location, and the other with participants examining a virtual painting on a clear white wall. The results showed that the combination of both data sources increased the number of tracked frames and allowed the system to work when the rear camera could not continue tracking.

The future work is aimed at utilizing both the rear and face cameras of the same device. Adapting a state-of-the-art library that provides the visual-inertial tracking will allow us to solve the main issues of our system, namely, a faster initialization of the tracking by utilizing the information from the other camera, and removing user's face from images obtained by the front camera. This will be followed by investigating which camera provide the best data for tracking, and optimizing the computation by removing the other camera from processing.

Acknowledgments

The authors wish to thank David Střelák for providing us with the model of the church, and all participants for evaluating the system. Part of this project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 739578 and the Government of the Republic of Cyprus through the Directorate General for European Programmes, Coordination and Development.

References

1. Apple: ARKit3 – Augmented Reality – Apple Developer (2019), `https://developer.apple.com/augmented-reality/arkit/`

2. Apple: Introducing multi-camera capture for iOS (2019), presented at Worldwide Developer Conference (WWDC 2019)
3. Bloesch, M., Omari, S., Hutter, M., Siegwart, R.: Robust visual inertial odometry using a direct EKF-based approach. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 298–304 (9 2015)
4. Concha, A., Loianno, G., Kumar, V., Civera, J.: Visual-inertial direct SLAM. In: 2016 IEEE International Conference on Robotics and Automation (ICRA). pp. 1331–1338 (5 2016)
5. Eckenhoff, K., Geneva, P., Bloecker, J., Huang, G.: Multi-camera visual-inertial navigation with online intrinsic and extrinsic calibration. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 3158–3164 (5 2019)
6. Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-scale direct monocular SLAM. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014. pp. 834–849. Springer International Publishing, Cham (2014)
7. Forster, C., Pizzoli, M., Scaramuzza, D.: SVO: Fast semi-direct monocular visual odometry. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). pp. 15–22 (5 2014)
8. Forster, C., Zhang, Z., Gassner, M., Werlberger, M., Scaramuzza, D.: SVO: Semidirect visual odometry for monocular and multicamera systems. IEEE Transactions on Robotics 33(2), 249–265 (4 2017)
9. Galatis, P., Gavalas, D., Kasapakis, V., Pantziou, G., Zaroliagis, C.: Mobile augmented reality guides in cultural heritage. In: Proceedings of the 8th EAI International Conference on Mobile Computing, Applications and Services. pp. 11–19. MobiCASE'16, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium (2016)
10. Gomez-Ojeda, R., Moreno, F.A., Zuñiga-Noël, D., Scaramuzza, D., Gonzalez-Jimenez, J.: PL-SLAM: A stereo SLAM system through the combination of points and line segments. IEEE Transactions on Robotics 35(3), 734–746 (6 2019)
11. Google: ARCore – Google Developers (2019), `https://developers.google.com/ar`
12. Heng, L., Lee, G.H., Pollefeys, M.: Self-calibration and visual SLAM with a multi-camera system on a micro aerial vehicle. Autonomous Robots 39, 259–277 (2015)
13. Hsieh, C.H., Lee, J.D.: Markerless augmented reality via stereo video see-through head-mounted display device. Mathematical Problems in Engineering 2015, 1–13 (2015)
14. Karrer, M., Schmuck, P., Chli, M.: CVI-SLAM – Collaborative visual-inertial SLAM. IEEE Robotics and Automation Letters 3(4), 2762–2769 (10 2018)
15. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality. pp. 225–234 (11 2007)
16. Lee, G.A., Dünser, A., Kim, S., Billinghurst, M.: CityViewAR: A mobile outdoor AR application for city visualization. In: 2012 IEEE International Symposium on Mixed and Augmented Reality - Arts, Media, and Humanities (ISMAR-AMH). pp. 57–64 (11 2012)
17. Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., Furgale, P.: Keyframe-based visual–inertial odometry using nonlinear optimization. The International Journal of Robotics Research 34(3), 314–334 (2015)
18. Li, P., Qin, T., Hu, B., Zhu, F., Shen, S.: Monocular visual-inertial state estimation for mobile augmented reality. In: 2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR). pp. 11–21 (10 2017)

19. Mourikis, A.I., Roumeliotis, S.I.: A multi-state constraint Kalman filter for vision-aided inertial navigation. In: Proceedings 2007 IEEE International Conference on Robotics and Automation. pp. 3565–3572 (4 2007)
20. Mur-Artal, R., Montiel, J.M.M., Tardós, J.D.: ORB-SLAM: A versatile and accurate monocular SLAM system. IEEE Transactions on Robotics 31(5), 1147–1163 (10 2015)
21. Mur-Artal, R., Tardós, J.D.: Visual-inertial monocular SLAM with map reuse. IEEE Robotics and Automation Letters 2(2), 796–803 (4 2017)
22. Oppermann, L., Blum, L., Lee, J.Y., Seo, J.H.: AREEF Multi-player underwater augmented reality experience. In: 2013 IEEE International Games Innovation Conference (IGIC). pp. 199–202 (2013)
23. Oskiper, T., Zhu, Z., Samarasekera, S., Kumar, R.: Visual odometry system using multiple stereo cameras and inertial measurement unit. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8 (6 2007)
24. Park, J., Seo, B.K., Park, J.I.: Binocular mobile augmented reality based on stereo camera tracking. Journal of Real-Time Image Processing 13, 571–580 (2017)
25. PTC: Vuforia: Market-Leading Enterprice AR (2020), `https://www.ptc.com/en/products/augmented-reality/vuforia`
26. Pumarola, A., Vakhitov, A., Agudo, A., Sanfeliu, A., Moreno-Noguer, F.: PL-SLAM: Real-time monocular visual SLAM with points and lines. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). pp. 4503–4508 (5 2017)
27. Qin, T., Li, P., Shen, S.: VINS-Mono: A robust and versatile monocular visual-inertial state estimator. IEEE Transactions on Robotics 34(4), 1004–1020 (8 2018)
28. Romero-Ramirez, F.J., noz Salinas, R.M., Medina-Carnicer, R.: Speeded up detection of squared fiducial markers. Image and Vision Computing 76, 38–47 (2018)
29. Schöps, T., Engel, J., Cremers, D.: Semi-dense visual odometry for AR on a smartphone. In: 2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR). pp. 145–150 (9 2014)
30. Seo, B.K., Kim, K., Park, J., Park, J.I.: A tracking framework for augmented reality tours on cultural heritage sites. In: Proceedings of the 9th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry. pp. 169–174. VRCAI '10, ACM, New York, NY, USA (2010)
31. Tribou, M.J., Harmat, A., Wang, D.W., Sharf, I., Waslander, S.L.: Multi-camera parallel tracking and mapping with non-overlapping fields of view. The International Journal of Robotics Research 34(12), 1480–1500 (2015)
32. Usenko, V., Engel, J., Stückler, J., Cremers, D.: Direct visual-inertial odometry with stereo cameras. In: 2016 IEEE International Conference on Robotics and Automation (ICRA). pp. 1885–1892 (5 2016)
33. Čejka, J., Zsíros, A., Liarokapis, F.: A hybrid augmented reality guide for underwater cultural heritage sites. Personal and Ubiquitous Computing (2020)
34. Vlahakis, V., Ioannidis, N., Karigiannis, J., Tsotros, M., Gounaris, M., Stricker, D., Gleue, T., Daehne, P., Almeida, L.: Archeoguide: An augmented reality guide for archaeological sites. IEEE Computer Graphics and Applications 22, 52–60 (9 2002)
35. Wikitude: Wikitude Augmented Reality: the World's Leading Cross-Platform AR SDK (2020), `https://www.wikitude.com`
36. Yang, S., Scherer, S.A., Zell, A.: Visual SLAM for autonomous MAVs with dual cameras. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). pp. 5227–5232 (5 2014)
37. Zhang, N., Zhao, Y.: Fast and robust monocular visua-inertial odometry using points and lines. Sensors 19(20) (2019)