Methodology and Implementation of an Architecture for Egocentric Manual Interactivity in Monocular Augmented Reality

1st Éverton C. Acchetta
FEI Comp. Science
São Bernardo, Brazil
eve.023@hotmail.com

2nd Lucas P. Laheras
FEI Comp. Science
São Bernardo, Brazil
lucaslaheras@hotmail.com

3rd Helmuth A. Risch
FEI Comp. Science
São Bernardo, Brazil
unifhfilho@fei.edu.br

4th Vinicius L. O. P. Santos
FEI Comp. Science
São Bernardo, Brazil
viniciuslops@hotmail.com

5th Paulo S. Rodrigues
FEI Comp. Science
São Bernardo, Brazil
psergio@fei.edu.br

***Abstract*—Investments in Augmented Reality (AR) have grown considerably in recent years. This advance is due to the increased use of AR in areas such as education, training, games and medicine. In addition, technological advances in hardware enable devices that, a few years ago, were unthinkable. A popular example is Microsoft Hololens 2, which allows the user to use their own hands as a means of interacting with an AR experience. However, a disadvantage from this device is its high cost due to several sensors. Thus, this project offers an AR architecture that uses only a monocular RGB camera as a sensor, allowing the user to interact with an AR experience using their hands to perform gestures similar to the Microsoft Hololens 2 architecture, where it is possible to handle a virtual object in the same way that a real object would be manipulated. The results obtained are promising, where the verification of the interaction of the hand with the virtual object worked in approximately 80% of the tests carried out, respecting the path defined by hand movement.**

***Index Terms*—Augmented Reality, Monocular, Object Recognition.**

I. Introduction

Augmented Reality (AR) is considered one of the fastest growing areas in Computer Vision today. Companies such as *Apple*, *Microsoft*, among others, have invested billions of dollars annually to advance the area [11], in order to revolutionize the way people consume and interact with content. Applications that use AR can be used in several areas, especially in games, education, medicine and human-machine interaction.

Studies, such as the one presented by [20], show that the use of AR in education results in greater student engagement, in addition to allowing a better understanding of subjects that require visualization of concepts. Also in the area of education, AR has been used for specialized training, allowing more people to have appropriate training in any location, without the need for real contact with the training object.

AR can be used with or without reference to artificial markers. Artificial markers are usually binary images, making them simple to locate. Despite facilitating the location, the system becomes dependent on markers for its operation. With the advancement in the area of computer vision, the use of methods such as locating surfaces and objects has been increasingly common.

Currently, according to the bibliographic research carried out for the development of this work, there are three types of interaction with a virtual object. Screen interaction, where the movement of the virtual object is performed using a mouse or the touchscreen [1]. Interaction with specific gestures, where some gestures are selected that, when recognized, activate a [4] function. Hand interaction, making the hand able to interact as if the virtual object were real [15], but with current technology this type of interaction ends up being dependent on many sensors.

Thus, this work proposes an architecture that allows the creation of an AR experience without ambient markers and using only a monocular Red Green Blue (RGB) camera as a sensor. This experience will allow the user to interact with the virtual object in a similar way to *Microsoft Hololens* 2, where the user can move (push, pull or lift) the virtual object, taking their own hand to where the virtual object is being represented in the real world and interacting with this object as if it were a real object. This architecture can be used as a new method of human-machine interaction for low cost hardware.

II. Related Works

In this section, we will present works related to the subjects that will be addressed in this work. With the intention of improving the understanding of the relationship of these articles with our proposed theme, the works were separated into 3 categories: Camera calibration techniques and their orientation in the environment; recognition of objects and surfaces through monocular, stereo and RGB-D images; and recognition of human gestures and body movements.

A. Camera Calibration Techniques and Their Orientation in the Environment

The work of [18] presents a tutorial for visual odometry techniques. The author set out to write a guide that contains techniques for camera model perspective, omnidirectional camera models, camera calibration, 2D-2D, 3D-3D and 3D-2D transformations, among others. The author seeks in his text to motivate the reader to formulate solutions for growth in the area.

In the work of [14], an improvement in monocular Simultaneous Localization and Mapping (SLAM) methodologies was proposed, where the authors couple to the process the ability to close cycles, reusing their mapping system to minimize deviations in locations already mapped. The technique was designed to work on monocular cameras. Experiments demonstrate that its monocular SLAM retrieves scale metrics with high accuracy, surpassing the state of the art in stereo visual-inertial odometry.

The work of [17] presents a large-scale image-based localization approach. The proposal allows to use points of interest that are most likely to be useful in 2D-3D conversion.. The technique presented efficient localization times in relation to the state-of-the-art. The authors demonstrate that co-visibility information, available in the Structure-from-Motion (SfM) process, can be used to speed up all stages of the localization process.

To estimate the distance between monocular cameras, [21] propose a framework to estimate both the distance and movement of cameras between unstructured video sequences. Using networks that contain poses with single and multiple views, the distance with the loss acquired in the image from the distortions close to the object is calculated. The authors conclude that, despite having achieved good results in their evaluations, the problem of automatic inference of structures in 3D scenes is still an open problem.

B. Object and Surface Recognition Through Monocular, Stereo and RGB-D Images

The work of [3] presents a method of visual odometry with sparse and direct precision, called Direct Sparse Odometry (DSO). The work combined probabilistic models and parametric optimization. The results demonstrated the superiority over state-of-the-art methods.

The works of [8], [6], [5], [7], [9] present competitions in the area of object recognizers, where it was possible to notice a dominance in the performance of recognizers based on deep learning techniques, mainly on CNNs. For the evaluation of each of the recognizers, a database created for each of the competitions was used, with the main evaluation factors being the accuracy of the tests and the speed with which the recognizer was able to find the object. It is possible to notice that the number of recognizers based on CNNs has grown every year, especially in the work of [9], where all competitors in the top 10 make use of such.

C. Recognition of Human Body Movements and Gestures

In the work of [12], the authors aimed to reconstruct a hand skeleton in a 3D environment from a monocular RGB sequence. The process combines a CNN with a kinematic 3D handheld model that is capable of producing results even under occluding conditions and varying viewpoints. It's also robust to hand movement so its hand anatomy remains realistic over each frame. The method demonstrates significant improvements compared to the similar work of [22], especially at the point where there is an occlusion in the scenario.

[19] use neural networks with long-term temporal convolutions (LTC), for learning video actions. They demonstrate that LTC-CNN models with increased time extensions improve the accuracy of stock recognition. The authors rely on two evaluation metrics: precision by frame, and precision by video. The authors highlight that the results using LTC_{Flow} algorithms in conjunction with average fusion RGB, outperformed the average baseline *Twostream* by 4.8% and 6.8% in the bases of UCF101 and HMDB51 data, respectively.

III. Methodology

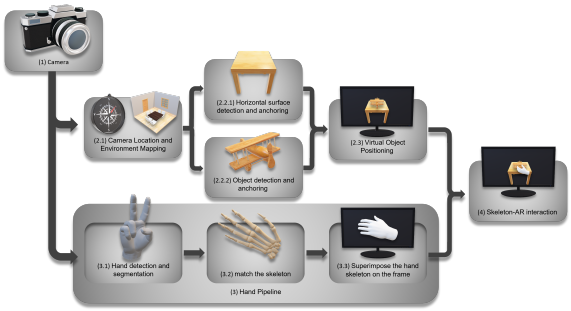

Fig. 1. Schematic diagram of the proposed methodology.

This section describes the flow of processing performed for the interaction of human hands with virtual objects projected in AR. This flow is composed of 8 main steps, illustrated in Figure 1. These steps will be explained in detail throughout the next sub-sections.

The process starts with Step (1), where images are obtained through a camera. After that, there are two workflows (2 (upper in Figure 1) and 3 (lower in Figure 1)) occurring in parallel.

The flow corresponding to Step (2) starts with a process of locating the camera and mapping the environment (Step 2.1), followed by two options (depending on the desired experience, a specific type of anchor must be used): The first (Step 2.2.1) consists of identifying horizontal surfaces and their uses as anchors; The second (Step 2.2.2) is about identifying previously trained objects and also their uses as anchors. Finishing this step, there is the positioning of the virtual object (Step 2.3).

Starting the flow of Step (3), there is the detection and segmentation of the hand (Step 3.1). After this process, the hand skeleton is matched with the result obtained previously (Step 3.2). Finally, there is the superimposition of the skeleton in the frame (Step 3.3).

Joining the two parallel flows, there is Step (4), responsible for the interaction of the skeleton obtained in step (Step 3.2) with the virtual object positioned in the scene in step (Step 2.3).

A. Camera

In this step, an RGB monocular image capture device was used. For this work, the rear camera of an Apple iPhone X smartphone was used.

B. Camera Location and Environment Mapping

In this step, the environment was mapped and the camera coordinates were determined using the ORB-SLAM algorithm [13], which uses points of interest detected in the environment.

Fig. 2. Distance Calibration Methodology for ORB-SLAM.

The ORB-SLAM, when used in its monocular mode, arbitrarily defines a scale for its coordinates, based on the distance between MapKeyPoints through the frames in the calibration process, making impossible to predict the exact value. Therefore, a distance calibration process was created, detailed in Figure 2.

In Step (1) of Figure 2, the standard initialization of the ORB-SLAM is performed, making sure that the camera is completely perpendicular to the ground (parallel to the y axis of the world). In Step (2), the camera is placed in a known location and the calibration process in the engine Panda 3D begins [2]. Finally, in Step (3), the camera must be moved one meter away from the starting position and then select the "end of calibration" option in Panda 3D.

This process is performed by calculating the Euclidean distance between the initial position $P_A = (x_A, y_A, z_A)$ and the final position $P_B = (x_B, y_B, z_B)$ of the camera through the Equation (1).

$$d_{AB} = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2 + (z_B - z_A)^2} \quad (1)$$

After obtaining the distance equivalent to 1 (one) meter, all coordinates received from the ORB-SLAM are now adjusted in all frames, following the Expression $P = \left(\frac{x}{d_{AB}}, \frac{y}{d_{AB}}, \frac{z}{d_{AB}}\right)$. With that, we have the coordinates system calibrated to meters.

In Figure 3 an environment initialized by ORB-SLAM is shown.

C. Horizontal Surface Detection and Anchoring

This step is considered an alternative to the step described in sub-section III-D, in which case it is responsible for detecting horizontal surfaces and specifying anchors on these surfaces, allowing the step described in sub-section III-E to take place.

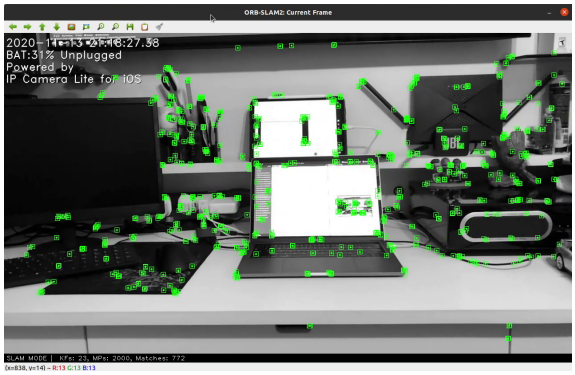

Fig. 3. ORB-SLAM with environment detected.

For the effective detection of the surface, the MapKeyPoints generated by the ORB-SLAM were used. The 3D coordinates of each of the MapKeyPoints visible in the frame are compared to each other, in order to group them so that the MapKeyPoints with similar heights (a variation was defined through empirical tests). acceptable of $+/-2$ centimeters between heights can be considered coplanar.

With these groups defined, the largest is selected, which corresponds to the plane with the largest amount of coplanar MapKeyPoints of the frame in question, thus translating into the largest surface at that moment. After selecting the group corresponding to the plan, only 4 (four) MapKeyPoints are selected:

- MapKeyPoint with the most negative x coordinate, defined as $M_{x1} = (x_1, y_1, z_1)$;
- MapKeyPoint with the most positive x coordinate, defined as $M_{x2} = (x_2, y_2, z_2)$;
- MapKeyPoint with the most negative z coordinate, defined as $M_{z1} = (x_3, y_3, z_3)$;
- MapKeyPoint with the most positive z coordinate, defined as $M_{z2} = (x_4, y_4, z_4)$.

Thus, we have the extreme points of the plane in question, just defining the 3D coordinate of the midpoint of this plane, through the expression $P = \left(\frac{x_1+x_2}{2}, \frac{y_1+y_2+y_3+y_4}{4}, \frac{z_3+z_4}{2}\right)$. The point P is then used as the origin coordinate for the plane generated by Panda 3D. This process only occurs when the user selects the option to detect horizontal surface in Panda 3D, in order to reduce system processing.

With the horizontal plane defined, the coordinate of the anchor is created, being defined as $A = (0, 0, 0)$ in relation to the created plane.

D. Object Detection and Anchoring

This step is an alternative to step (2.2.1). In order to use YOLOv4 [16] for object detection, a training base (here called YOLO Base) was used, consisting of five different objects, with 150 (one hundred and fifty) photos each, totaling 750 (seven hundred and fifty) photos. These photos were captured from various angles and possible poses in at least 3 (three) different locations.

After YOLO training, the flow illustrated in Figure 4 is executed, where in sub-step (1), the camera is pointed at the object to be identified. In sub-step (2), there is the bounding box generated by YOLO, in addition to the identification of the detected object. Thus, both the location and the object label are defined for each frame.

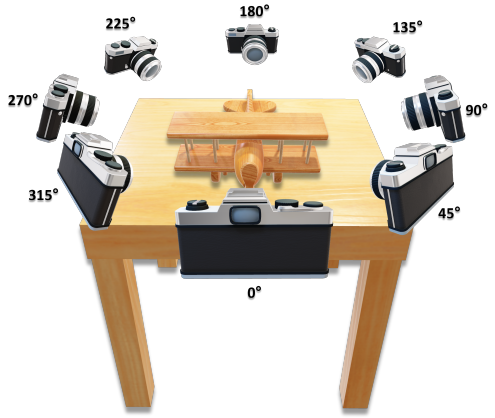

Fig. 4. Object detection flow and its orientation in relation to the camera.

Thus, the angles and directions of each object in relation to the camera pose are then calculated, as illustrated in Figure 5, where the first angle is the β, which indicates the position of the camera in the horizontal plane in relation to the front of the object. The second is the γ, indicating the camera angle in the vertical plane, in relation to the horizontal plane.

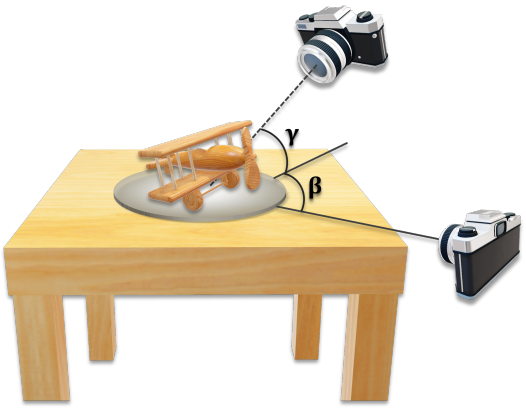

Fig. 5. Illustration of camera angles in relation to the object.

To define the angle of the object a homography-based approach was used. The database used to find the angulation (called here Angular Base) consists of image captures of objects at the following angles β in the horizontal plane (in Figure 5): 0°, 30°, 60°, 90°, 120°, 150°, 180°, 210°, 240°, 270°, 300° and 330°. For each horizontal angle, pictures are taken at the following angles γ in the vertical plane (in Figure 5): 0°, 15°, 30°, 45° and 60°, totaling 60 photos per object. These photos were captured with the objects on a green background, in order to minimize the amount of fiducial points that are not part of the object.

Initially, the SIFT descriptor [10] is applied to each image of the angular base, in order to extract its fiducial points. As the name of the images are labeled containing the object name and the angle (β and γ), it is thus possible to identify the fiducial pattern of each object at all angles considered.

In sub-step (3) of Figure 4, the fiducial points contained in the bounding box generated by the YOLO detection of the input frame are extracted and in sub-step (4) the data are serialized for comparison.

Thus, each point will be compared with the points of the base, as illustrated in sub-step (5) of Figure 4, in order to identify the set that has the greatest amount of fiducial points of the frame under analysis, determining the angle of the object in relation to the camera.

To anchor the virtual object in the virtual world, the position of the real object in relation to the camera is estimated, using directly proportional quantities for x and z in the virtual world, and angular diameter for y, representing the depth in the virtual world. For this method to be possible, it is necessary that all images maintain a specific aspect ratio of 16:9. Thus, the directly proportional magnitude is calculated, using the values of the width and height of the real object and the normalized bounding box of the object in the frame. In angular diameter, the apparent angle θ that an object is viewed is related to the actual size of the known object d and the distance to the camera D (Equation (2)). The angular diameter is obtained from the object's width in the frame, which has a known angular aperture. This method was chosen because it is simple and quick to calculate, aiming at better performance.

$$\tan(\theta) = \frac{d}{D} \tag{2}$$

E. Virtual Object Positioning

The position of the virtual object mainly depends on the surface or object detection step (sub-sections III-C and III-D). If the user chooses the surface option, the center of the object will be positioned in the environment at the same coordinate as the anchor of the plane with the height (z dimension) added with the distance from the bottom of the object to the center, thus positioning so that the plane and the bottom of the object just touch each other. If the object detection option is choosen, the virtual object will be positioned at the same coordinate of the calculated anchor in sub-section III-D.

F. Hands pipeline

This pipeline was performed by the algorithm proposed by [12], thus allowing the detection and segmentation of the hand in the camera frame to be performed (3.1 of Figure 1), corresponding it immediately afterwards to a 3D skeleton composed of 21 movable joints (3.2 of Figure 1) that are used for the virtualization of the hand in the frame (Step 3.3 of Figure 1).

G. Skeleton's Interaction with Augmented Reality

The last step performs the union of all the steps previously performed, in order to allow the user to interact with a virtual object (projected in the augmented reality environment) using their own hand.

First, the 21 coordinates of the articulations of the hand skeleton, obtained in Section III-F, are stored at each frame. Thus, these coordinates are used to position a 3D mesh of a human hand in the same position in space as the user's hand. This mesh is transparent, so it only serves as a representation of the user's hand in the virtual world, in order to allow the next sub-step to be possible.

Finally, the collision interaction of the 3D mesh of the user's hand with the 3D mesh of the virtual object is performed. This is accomplished using the engine Panda3D. Each object studied has a set of primitive geometries (spheres, cubes or cylinders) for collision calculations, as well as the 3D mesh of the user's hand. This allows the collision processing between meshes to be less than considering all the geometry of the objects. When detecting an intersection between the meshes, the objects are moved by contact with the 3D mesh of the hand, thus allowing them to be pushed and even lifted, when there are contacts that allow these movements.

Figure III-G shows the model related to the human hand in contact with the virtual model.

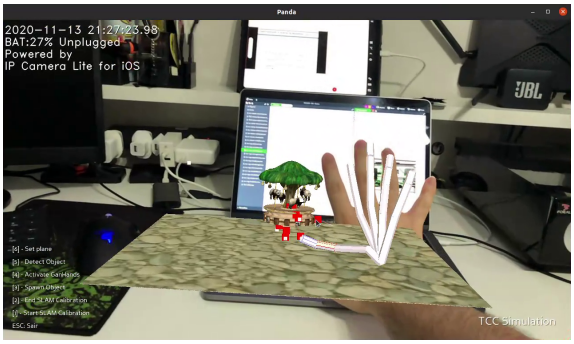

Fig. 6. Red squares indicate hand contact with the virtual model. The surface does not interact with the hand.

IV. Results

In order to measure the reliability of this work, five categories of experiments were proposed, with the aim of validating the steps of the proposed methodology. These categories are divided into two stages: experiments with isolated modules (classification of horizontal surfaces, detection of defined objects and detection of the angle of objects) and experiments with full implementation (definition of the resolution of the camera used and interaction of the virtual object with the movements of hand).

For the surface classification results, 15 experiments were performed, were the camera was pointed at 2 surfaces (a desk and a bed, one at a time), where the camera was moved in a semi-circle shape, with the intention of map the surface from different angles. From this sample, 13 experiments were considered as successful and the other 2 experiments were considered as failure, as they did not meet the necessary requirements. Therefore, as 13 out of 15 experiments were considered successful, this test was accurate to approximately 86.67%.

For the verification of the surface detection experiment, the mean assertiviness was used, comparing the results given by the algorith and if the image was really showing a surface. For the YOLO detections, each of the 5 items (150 images each) that were previously captured were sent to YOLO and its assertiveness percentages and IOU were measured after training. All objects in the experiment scored correctly with a 100% true positive rate. Table I presents the results regarding the mean IoU, standard deviation and variance for each object and for all objects considered.

TABLE I
TABLE LISTING THE MEAN ASSERTIVENESS, STANDARD DEVIATION AND VARIANCE OF EACH OBJECT

	Mean IOU	Standard deviation	Variance
Small Car	93,56%	3,85%	0,1480%
DualShock4	90,57%	5,27%	0,2776%
IamGroot	95,23%	1,87%	0,0349%
MiniCraque	93,67%	3,02%	0,0911%
PlayStation2	91,86%	3,66%	0,1341%
All objects	**92,98%**	**4,04%**	**0,1630%**

In the quantitative object angle detection experiments, where the number of fiducial points found within the bounding box provided by YOLO was compared with the reference points of the base used for YOLO training, with predefined angles (totaling 150 images each item). The results of this test are represented in Table II.

TABLE II
TABLE OF QUANTITATIVE TESTS, RELATING IMAGE RESOLUTIONS TO OBJECTS, DEMONSTRATING THE NUMBER OF IMAGES THAT WERE SUCCESSFULLY IDENTIFIED

	640x480	800x600	960x720	1280x960	1440x1080
Small Car	102	124	145	147	150
DualShock4	20	37	66	127	139
IamGroot	37	64	82	112	115
MiniCraque	53	71	94	130	136
PlayStation2	80	115	138	149	149

For the resolution tests, 1080p, 720p and 640p resolutions were used. After the tests, the average frame rate of the system was calculated. The results were: 47 fps for 640x480 resolution, 35 fps for 1280x720 resolution and 18 fps for 1920x1080 resolution.

In the experiments carried out to validate the interaction of the hand with the object (illustrated in Figure 7), the precision of the interaction between the hand mesh and the virtual object projected in the scene was taken into account. Thus, starting and ending points were defined for the object to be pushed. A total of 15 tests were performed, where 12 times in which this simulation was performed, the object reached the end point covering the pre-defined 40 centimeters, respecting the path defined by the movement of the hand. In 3 of these situations,

Fig. 7. Visualization of the interaction between the mesh of the real hand and the virtual object, which in this case is a model of a wrench. (a) Hand almost touching object. (b) Hand tangent to object. (c) Hand pushing object.

the object did not successfully reach the destination. Thus, the accuracy of this test was 80%.

V. Conclusion

From the bibliographic research, it was verified that most of the works used hardware resources, such as depth sensors, stereo cameras, RGBD cameras, touchscreen, accelerometer and gyroscope to perform an interaction with the virtual object in augmented reality. This work successfully performed this interaction using a monocular RGB camera, where all the environment mapping, hand detection, object detection and interaction computation are performed via software in a fluid way (35 fps), but it can be noted in some results, the accuracy with limited hardware ends up being lower due to the fact that GANHands has the flaw where the thumb generates unreal movements, moving the object in an unwanted way.

For future work, there are some improvements and even new implementations, such as: Improvement of the algorithm for hand recognition and structuring of the mesh in the 3D environment; Addition of virtual objects that allow a more immersive interaction with the user; Application of this work to new contexts of study, such as medicine, training, remote support and schooling.

Acknowledgment

We would like to thank CNPq (Project 155130/2019-6), as well as the FEI (Inaciana Educational Foundation) and LNCC (National Scientific Computing Laboratory) for supporting this work.

References

[1] Matthias Baldauf, Katrin Lasinger, and Peter Fröhlich. Private public screens: Detached multi-user interaction with large displays through mobile augmented reality. In *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia*, MUM '12, New York, NY, USA, 2012. Association for Computing Machinery.

[2] Disney. Panda3d, 2019.

[3] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:611–625, 2016.

[4] Ahmad Karambakhsh, Aouaidjia Kamel, Bin Sheng, Ping Li, Po Yang, and David Dagan Feng. Deep gesture interaction for augmented anatomy learning. *International Journal of Information Management*, 45:328–336, 2019.

[5] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Cehovin Zajc, Tomas Vojir, Gustav Hager, Alan Lukezic, Abdelrahman Eldesokey, et al. The visual object tracking vot2017 challenge results. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1949–1972, 2017.

[6] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Čehovin Zajc, Tomáš Vojír, Gustav Häger, Alan Lukežič, Gustavo Fernandez Dominguez, Abhinav Gupta, Alfredo Petrosino, Alireza Memarmoghadam, Alvaro Garcia-Martin, Andrés Montero, Andrea Vedaldi, Andreas Robinson, Andy Ma, Anton Varfolomieiev, and Zhizhen Chi. The visual object tracking vot2016 challenge results. volume 9914, pages 777–823, 10 2016.

[7] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka ˇCehovin Zajc, Tomas Vojir, Goutam Bhat, Alan Lukezic, Abdelrahman Eldesokey, et al. The sixth visual object tracking vot2018 challenge results. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[8] Matej Kristan, Jiri Matas, Ales Leonardis, Michael Felsberg, Luka Cehovin, Gustavo Fernandez, Tomas Vojir, Gustav Hager, Georg Nebehay, and Roman Pflugfelder. The visual object tracking vot2015 challenge results. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 1–23, 2015.

[9] Matej Kristan, Jiri Matas, Ales Leonardis, Michael Felsberg, Roman Pflugfelder, Joni-Kristian Kamarainen, Luka Cehovin Zajc, Ondrej Drbohlav, Alan Lukezic, Amanda Berg, et al. The seventh visual object tracking vot2019 challenge results. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[10] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[11] Tim Merel. Digi-capital: Over $4.1 billion invested in ar and vr in 2019, Mar 2020.

[12] Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. Ganerated hands for real-time 3d hand tracking from monocular rgb. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 49–59, 2018.

[13] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[14] Raúl Mur-Artal and Juan D Tardós. Visual-inertial monocular slam with map reuse. *IEEE Robotics and Automation Letters*, 2(2):796–803, 2017.

[15] N. O-larnnithipong, N. Ratchatanantakit, S. Tangnimitchok, F. Ortega, and A. Barreto. Hand tracking interface for virtual reality interaction based on marg sensors. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 1717–1722, 2019.

[16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[17] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1744–1756, 2017.

[18] Davide Scaramuzza and Friedrich Fraundorfer. Visual odometry [tutorial]. *IEEE robotics & automation magazine*, 18(4):80–92, 2011.

[19] G. Varol, I. Laptev, and C. Schmid. Long-term temporal convolutions for action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1510–1517, 2018.

[20] Hsin-Kai Wu, Silvia Wen-Yu Lee, Hsin-Yi Chang, and Jyh-Chong Liang. Current status, opportunities and challenges of augmented reality in education. *Computers Education*, 62:41 – 49, 2013.

[21] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 340–349, 2018.

[22] Christian Zimmermann and Thomas Brox. Learning to estimate 3d hand pose from single rgb images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4903–4911, 2017.