2022 IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE) | 978-1-6654-8574-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/MetroXRAINE54828.2022.9967592

An easy Hand Gesture Recognition System for XR-based collaborative purposes

1st Nicola Capece, 2nd Gilda Manfredi, 3rd Vincenzo Macellaro
Department of Mathematics, Computer Science, and Economics
University of Basilicata, Potenza, Italy
nicola.capece@unibas.it, gilda.manfredi@unibas.it, vincenzo.macellaro@studenti.unibas.it

4th Pietro Carratù
Youbiquo S.r.l.
Cava de' Tirreni, Italy
pietro.carratu@youbiquo.eu

***Abstract*—In the last few years, collaborative tools have increased dramatically, partly to compensate for the distance between people due to the pandemic and partly to allow activities (work, entertainment, free time, *etc.*) to be carried out among people without having to worry about geographical distances. In this scenario, it was necessary to overcome the classic remote meeting tools (*e.g.* video, audio, chat), which have a reduced sense of presence. Extended Reality (XR) represents a Computer Graphics (CG) based innovative technology particularly suited to this purpose. Indeed, XR aims to develop Virtual Reality (VR), Augmented Reality (AR), and Mixed Reality (XR) solutions that can transform how people interact by increasing the sense of presence. This last aspect depends not only on the virtual/real objects and scene visualization but also on their interaction. In this regard, human-computer interaction (HCI) techniques represent a possible solution. However, these techniques depend on specific devices, such as Head Mounted Display (HMD), Smart Glasses, Depth and Tracking Cameras, *etc.*, whose costs make access difficult. For this reason, we propose a Hand Gesture Recognition (HGR) system that can be used in XR applications using a simple RGB camera. Our is a deep learning system based on MediaPipe, the state-of-the-art (SOTA) for hand tracking through simple RGB images [1], [2].**

***Index Terms*—Hand Gestures, Deep Learning, Human-computer Interaction**

I. Introduction

The need to reduce distances among people to ensure a high sense of presence has become one of the biggest challenges in computer science and CG. In the last years, more and more collaborative tools such as Mozilla Hubs, VRChat, and other general purposes browser-based web 3D tools [3] were developed and made accessible for users. Some of them ensure user presence by allowing the creation of personalized avatars [4], [5]. Generally, these tools guarantee high performance in terms of a sense of presence when scene visualization and interaction are performed using HMDs and their controllers. However, more recent HMDs are suitable for being combined with hand tracking devices, such as the Leap Motion controller, to allow interaction using freehand [6] and gesture recognition [7]. Furthermore, the last HMDs, such as Oculus Quest 2 and Vive Focus 3, integrate on-board hand-tracking sensors, capturing hand gestures directly and ensuring greater ease of handling and ergonomics, increasing the sense of presence, body ownership, embodiment, and agency [5], [8] of the users. As these devices are not accessible to all users due to their cost and usability, new technologies have recently emerged that allow freehand and gesture recognition through general-purpose devices. In this context, we propose a deep learning approach that allows HGR to use a simple, low-cost RGB camera.

In particular, we defined a system based on a well-structured pipeline in which the landmarks predicted from the MediaPipe Hands solution [2] are used as input to a simple feed-forward neural network (FFNN). The main advantage of our approach is that MediaPipe needs only a simple RGB hands-content camera frame as input to predict hands-landmarks. They can be obtained in real-time and are entirely independent of the camera features. The predicted landmarks correspond to the hand landmarks from which our FFNN can predict the corresponding hand gesture. To train our FFNN, we defined a dataset based on a dictionary of 15 gestures represented by many combinations of open and closed fingers and hands. Gestures are divided into static and dynamic based on their behavior. The former consisted of the FFNN predicted gesture, and the latter were obtained using the FFNN prediction as an activation gesture and performing assertion on the hands-landmarks for the subsequent frames.

We used our system for collaborative real-time 3D scene interaction in the XR environment using the well-known Unity 3D game engine and its network library called Netcode [1]. The proposed 3D application is a simple authoring tool use case helpful in building 3D scenes in real-time through collaborative multi-user interaction. Since our system is trained with landmarks captured in the egocentric mode [9], it can be used with smart glasses as an on-device system.

The remainder of this paper is structured as follows: in Section II an overview of related work was provided; in Section III we provide a background and motivations; in Section IV we described our proposed HGR system; A XR collaborative authoring use case is described in Section V, and finally the conclusions suggestions for future works were provided in Section VI.

II. Related Work

The 3D interaction with the hands benefits significantly from using the HGR system [10]. The first step to having a working HGR system is to choose the correct data acquisition

[1]https://docs-multiplayer.unity3d.com/netcode/current/about

978-1-6654-8574-6/22/$31.00 ©2022 IEEE

method to perform a determined task. In the literature, there are different data acquisition approaches characterized by using sensor gloves, markers, and hand images. The gloves approach uses sensors that convert the hand motion into electrical signals, which a computer must elaborate on. These signals give information about the fingers' coordinates and the hand configurations [11], [12]. These devices can be very precise, but they can be costly, heavy, and not very portable. On the other hand, the marker methods can be considered more straightforward, far less costly, and less heavy than sensor gloves. These techniques are based on the use of gloves with markers [13] or colored gloves [14] to locate the palm and fingers. In detail, a simple RGB camera can be used to identify the static positions of the markers or the color regions linked to the palm and the fingers. With marker methods, it is easier to retrieve the information about the finger position, but the camera cannot obtain all the information about the orientation because markers cannot be visualized for every hand position. Also, the marker methods are sensitive to light and background. For these reasons, new approaches based on bare hand detection have been introduced. Some studies focus on the reconstruction of the 3D hand model to detect the hand [15]–[17]. Because having an accurate 3D reconstruction can be computationally expensive, new methods for detecting and analyzing the hand skeleton have developed. Skeleton-based solutions adopt different techniques, such as multiple RGB cameras [18], RGB-Depth sensor [19], or a single RGB camera and two neural networks [2], to detect the hand skeleton. Skeletal features can be used for gesture recognition tasks. An example is a work proposed by F. Yang *et al.* [20] which adopted a Double-feature Double-motion Network to make the skeleton-based gesture recognition model smaller and faster. A. Caputo *et al.* [21] proposed a novel dataset formed by static, dynamic, and fine gestures for the skeleton-based HGR task. In recent years many virtual collaborative solutions have been adopted in different applications fields (*e.g.* medical and educational fields). Each collaborative system has a different method for interaction with the virtual scene based on the task to be performed. For example, C. Vuthea *et al.* [22] chose head-mounted displays and their controllers to interact with a virtual reality environment for liver surgery planning. These devices are also employed by N. Capece *et al.* [23] in a virtual reality application that permits users to collaborate in the creation of a 3D environment with meshes and lights. Other approaches, such as T. Piumsomboon *et al.* [24] use optical hand tracking instead of controllers to allow gesture recognition.

III. Background and Motivations

Our work aims to provide an innovative and accessible system that allows users to interact with collaborative 3D scenes in the XR environment using their hands. Since most of these interaction systems are based on specific devices such as Leap Motion, Microsoft Kinect, or other RGB-D cameras, we investigated the use of Deep Learning methods to disengage from them, making the system accessible to everyone. However, the use of these specific devices and the corresponding output [19] is well-structured to provide an optimal hand tracking [25] which is basically for the HGR [26]. Indeed, in recent times, many deep learning-based approaches have shown how it is possible to obtain good hand tracking using simple RGB webcams [27], [28] onboard on most PCs, smartphones, and wearables. One of the most promising and suitable approaches for our purpose is MediPipe Hands [2], which provides optimal documentation and a set of well-structured hand landmarks. MediaPipe consists of lightweight Deep Learning models [1] which can also be used on devices with limited computing resources. Hand tracking in real-time allows us to understand where the user's hands and their landmarks are positioned to the reference system of the 3D scene, providing information for the scene interaction (*e.g.* manage collisions, movements, *etc*). As shown in Figure 1, the MediaPipe Hand tracking solution uses a pipeline consisting of two models: a palm detector and a hand landmark model. The first locates the palm defining a bounding box oriented with the hand using the current RGB frame. Such a frame is cropped based on the predicted bounding box to reduce the need for data augmentation and allow the other model to be more precise in the hand landmarks localization. The Palm Detection Model, which is based on BlazeFace [29], is used on the first frame or when the hand prediction is lost, while for the other frames, the landmarks prediction of the previous frame is used to derive the palm bounding box, decreasing the computational effort. The hand landmark model is used to predict from the cropped frame 21 hand-knuckle coordinates (see Figure 4) detected inside the hand regions through regression. These coordinates consist of components of x, y, and z. The latter is obtained using the relative depth concerning the wrist, and for this reason, the coordinate spaces are indicated as $2.5D$. The use of our interaction system in the XR collaborative authoring tool is due to different research issues which are unsolved with classic interaction systems, such as 6-DOF controllers, gamepad, *etc.*. Such issues are the sense of embodiment, presence, body ownership, and the sense of agency. The first two feedback include the sense of self-location due to using the user's own hands and specific gestures to interact with the 3D scene. Another critical feedback is the body ownership associated with the visual hand's appearance and can be obtained mainly in the context of AR and MR. Finally, hand gestures ensure the sense of agency, related to the feeling of user action control [30]. To develop the XR and collaborative authoring purposes, we used the well-known Unity 3D game engine, which supports several XR platforms. To implement our use case, we used OpenXR, and the multi-user feature was implemented using the Netcode library. This mid-level networking library allows us to manage high-level protocols and networking frameworks. Indeed, Netcode is very useful for managing the concurrence, permissions, and ownership of scene objects, also called gameObjects (Unity 3D term), using basic RPC (Remote Procedure Call).

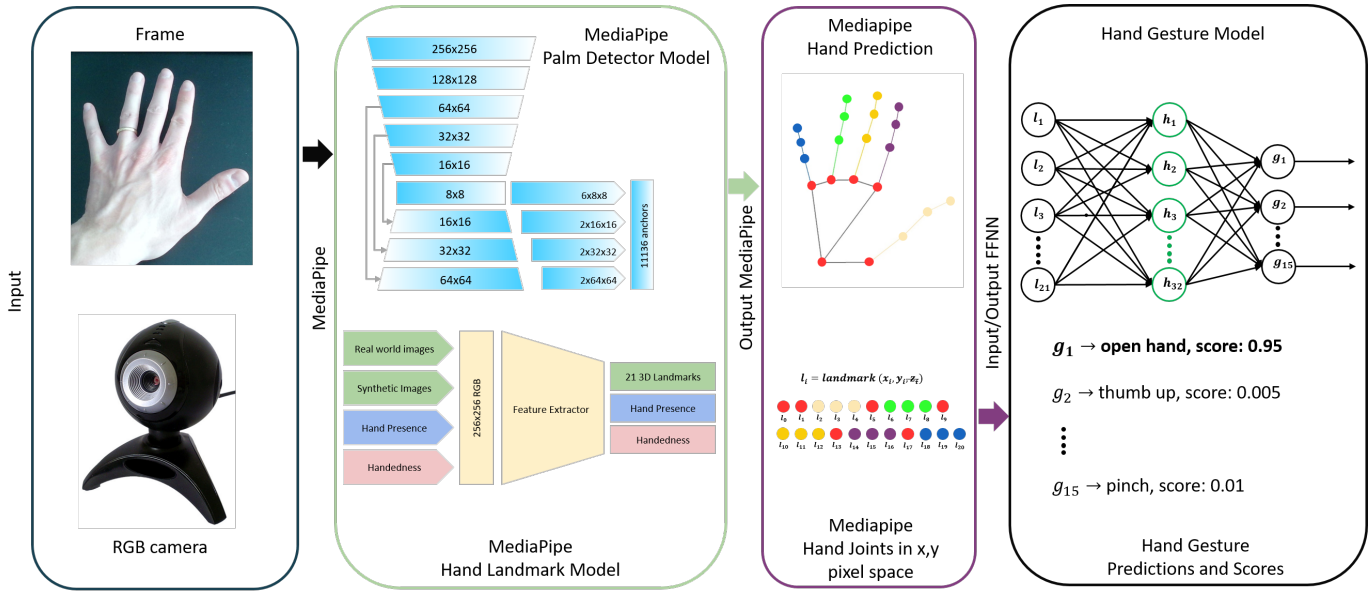

Fig. 1: HGR pipeline: The first block represents the current input RGB frame captured using an RGB camera. The input is processed with the MediaPipe Palm Detector Model and the Hand Landmark Model to predict the 2.5 hand-knuckle coordinates. Each coordinate's x and y components are processed from our FFNN to predict the current Hand Gesture.

IV. Hand Gestures Recognition

We defined a dictionary of 15 hand gestures based on their possible use in an interactive 3D environment. These gestures are derived from the state-of-the-art [21], [31], [32], and are represented by several combinations of open and closed fingers and hands. As shown in Figure 1, the input of our approach is represented by the 21 hand landmarks predicted from MediaPipe. Each landmark consists of x, y, and z coordinates (see Section III), but we used only the coordinates x, y, $2D$ ignoring the coordinate z, which is more useful for tracking. We used these components of the hand landmark model, which are defined in the pixel space as input to a simple feedforward neural network (FFNN) with a single hidden layer of 32 units (see Figure 1). The choice of the units' number was made by keeping a trade-off between computational cost and real-time performance. Furthermore, since the complexity of our network is low, we decided to not use the hyper-parameters optimization methods [33] as we have quickly and easily obtained the best configuration. We use a rectified linear unit (ReLU) activation function on the hidden layer because it is considered the better choice with respect to the classical sigmoid and hyperbolic tangent [34] allowing our FFNN to learn faster and perform better. On the last FFNN layer we used a Softmax [35] activation function.

Although several approaches treat dynamic gestures using data sequences such as recurrent neural networks (RNNs) [20], [21], we decided to use a simple FFNN for several reasons, including *(i)*: RNNs and other data sequences based approaches need to process the entire data sequence to provide a prediction. Such an approach cannot be used to predict a gesture and ensure that the associated action is performed in real-time; *(ii)*: the use of an RNN and similar assume that the dynamic gestures are asynchronous and therefore predicted only after the gesture has taken place; *(iii)*: XR collaborative applications are well-suited for synchronous dynamic gestures. The use of our simple FFNN model maintains the real-time performance of the whole process, reducing the computational impact. Our proposed gestures are divided into two templates: 8 static gestures and 7 dynamic gestures. Static gestures shown in Figure 2 are represented by a fixed hand pose directly predicted by our FFNN. In particular, each set of coordinates provided by the MediaPipe framework and based on the current camera frame are used as input to our FFNN. We consider the gesture represented from its output as valid.

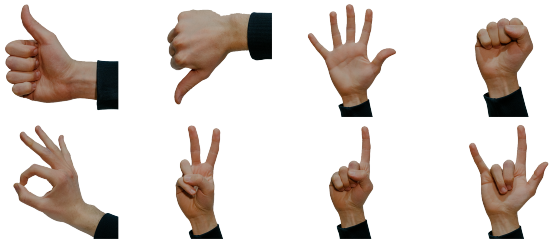

Fig. 2: Static gestures: thumb up, thumb down, open hand, closed hand, ok, pace, open index finger, rock.

The dynamic gestures shown in Figure 3 are characterized by moving hands. The dynamic gestures can be further divided into single and combo gestures based on the number of tracked hands. These gestures can be activated by a different starting-hand pose, keeping track of specific landmark positions. In particular, we used specific static activation gestures - that have been provided by our FFNN similarly to static gestures - and we have made assertions on the landmarks predicted by MediaPipe on the following frames to detect the type of dynamic gesture performed by the user.

A single gesture is detected using a single hand and its pose

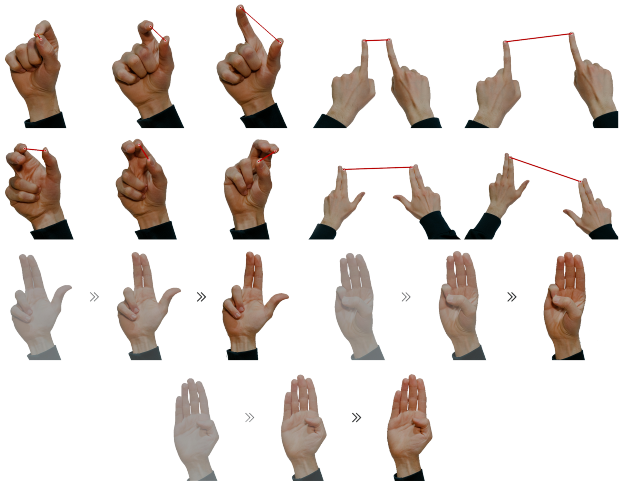

Fig. 3: Single dynamic gestures: pinch, rotation, two-fingers swipe, three-fingers swipe, and four-fingers swipe. Combo dynamic gestures: rotation (two hands), pinch (two hands).

as an activation gesture (see Figure 3). In detail, there are two, three, and four-finger swipe gestures, pinch, and rotation gestures. For the swipe gestures, once the FFNN predicts their corresponding activation gesture, it is possible to push a set of subsequent hand poses in a small stack to establish the movement's direction. For each of these poses, it is possible to consider one of the landmarks related to a hand phalanx and assess if its x component is greater than the previous to consider movement in the right direction and vice versa. In this way, we can distinguish between left- and right swipes, increasing the number of dynamic gestures. We consider the distance between the index-knuckle distal phalanges and the thumb-knuckle distal phalanges of subsequent hand poses using a stack to detect the pinch-in and pinch-out behavior for the single-hand pinch gesture. When this distance increases, it is a pinch-in; otherwise, it is a pinch-out. The single-hand rotation gesture is defined by considering two segments having a common endpoint and measuring the angle between them in real-time to establish whether it is a clockwise rotation, if this angle increases, or counterclockwise if it decreases. Combo dynamic gestures are detected using two hand poses to perform two FFNN inferences, one for each hand. For the combo pinch, when the initial hand poses are detected, the distances and positions of the distal phalanges of the index finger are used to establish the intensity of the gesture (see Figure 3). The combo rotation is similar to the single rotation, but the segment was computed between both hands' two index-knuckle distal phalanges.

To train our FFNN model, we developed a specific dataset consisting of 130000 hand pose samples taken with different hands and cameras and manually labeled. We split this dataset after an initial shuffle as 20% validation-set and the remaining as training-set. The testing was performed in real-time after the training. In particular, we used the RGB sensor of the Intel RealSense D455 camera with 1280×720 as resolution, the 40MP Huawei P30 back camera, and the 1080p MacBook

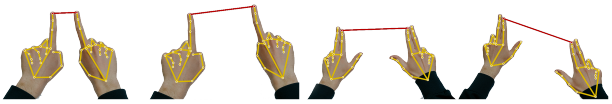

Fig. 4: Skeleton tracked by MediaPipe on both hands during pinch and two-handed rotation.

Pro (M1 Pro) camera. The samples are distributed equally among gestures. Each sample consists of 42 elements: the x and y values for each of the 21 hand landmarks provided from MediaPipe. As MediaPipe also allows egocentric hand tracking, the samples are taken and labeled in egocentric and non-egocentric modes. In this way, our approach can be used with a simple RGB camera placed in front of the user or XR devices such as smart glasses (with an RGB camera onboard) in an egocentric view. The FFNN was trained for 2000 epochs with Adam optimizer [36] algorithm and 0.0001 as the learning rate, obtaining a prediction accuracy of 98%.

V. Collaborative Authoring

The proposed HGR system was designed and developed to be used in a collaborative multi-user XR 3D environment. Each proposed gesture is associated with a specific action in an XR 3D environment. As explained in Section III we validated our approach in a Unity 3D game engine environment taking advantage of its Netcode mid-level networking library to allow the multi-user scene authoring. For the testing purpose, such an application was distributed among two clients equipped with a simple RGB camera to track the hands' landmarks and predict the gestures. When the user clients are connected to the scene, they can interact with visible gameObjects using hand gestures. As shown in Table I we implemented a set of user interaction actions associated with our designed hand gestures. Our collaborative scene is empty when client users connect for the first time; later, they can add or delete virtual objects. They can be selected or deselected from a special menu that can be opened or closed with the "Pace" gesture. The objects can be selected and deselected from the menu or scene by tapping them using the "Open Index Finger" gesture. The user can navigate the menu via the "Three Fingers Swipe" gesture, which allows it to be scrolled up or down. The selected object can be added to the scene through the "Thumb Up" gesture and deleted through the "Thumb Down" gesture. The user will keep the selection on the current added object and have its property. All the objects selected by the user will have his ownership, which can be acquired by another client or passed and kept by the server. When the user deselects an object in the scene, its ownership is transferred to the server, making it available to be acquired by another client, ensuring its concurrency. The client user can select, deselect, add, and remove more than one object, creating an object's group that will behave as a single object. Group objects can be translated with the "Pinch In" and released with the "Pinch Out" gestures. On the other hand, an object belonging to a group can be individually grasped and translated through the "Closed Hand" gesture and released through the"Open Hand"

Gesture	Action
Thumb Up	Add the selected objects from the menu into the scene
Thumb Down	Remove the selected objects from the scene
Open Hand	Release an object
Closed Hand	Grab and move a scene object
Ok	Confirm changes
Pace	Open/Close objects menu
Open Index Finger	Select an object on the menu or scene
Rock	Release all selected objects
Pinch	Grab, move, and drop an object group in the scene
Rotation	Rotate the single selected object
Two Fingers Swipe Right/Left	Undo/Redo
Three Fingers Swipe Right/Left	Scrolls up/down the objects menu
Four Fingers Swipe Right/Left	Select the next/previous material of the selected object
Combo Rotation	Rotate groups of selected objects
Combo Pinch	Zoom In and Zoom Out of the scene objects

TABLE I: User interaction actions are associated with hand gestures. The direction of the two and four fingers swipe gestures is irrelevant, while the three swipe gesture direction allows the up/down scrolling of the objects menu.

gesture. Likewise, the single object belonging to a group can be rotated using the "Rotation" gesture, and the group can be rotated using the "Combo Rotation" gesture. The single or group of objects can be zoomed in and out using the "Combo Pinch" gesture. It is possible to change some components, such as the materials of the objects, by selecting them through the "Four Finger Swipe left/right" gesture, scrolling through the next and the previous one. Furthermore, all objects can be deselected with a "Rock" gesture. Finally, we implemented two outline functionalities of the application, such as the "Ok" gesture to confirm changes and the "Two Finger left/right" gesture to perform undo/redo options. In Figure 5 the scene in desktop mode is shown for example purposes, using AR to visualize hand tracking. Hand movement is reported for also debugging in the 3D scene using a specific representation. However, this scene can also be viewed in virtual, AR, and MR using a smartphone and HMD or a simple Google Cardboard. Our gesture recognition system can also be used in the egocentric mode, as explained in Section IV since it has also been trained with hands tracked in this way.

VI. Conclusions

We propose an HGR system in which both hand landmarks and gestures are predicted from two neural networks. The first is the MediaPipe Hands solution, which predicts hand landmarks from simple RGB images. The second is an FFNN which takes the predicted landmarks and provides the corresponding hand gesture. We implemented two types of gestures: statics, and dynamics. The first are represented by the direct prediction of our FFNN, while the second are obtained also considering the position of the hand landmarks and their movements. Finally, we implemented a simple multi-user XR collaborative authoring tool based on Unity 3D, OpenXR, and Netcode to validate the effectiveness of our system. Our approach can also be used on the XR web browser environment. Given the multiplatform suitability of the web browser and the CG application support, the XR environment can also be approached using mobile devices (*e.g.* smartphones and smart glasses) that allow for a reasonably immersive feeling (*e.g.* through cardboard). In the future, we will conduct a user study to evaluate users' learning curve for hand gestures and the usability and sentiment of our system in the XR environment. One of the main challenges that must be addressed in future developments is to determine which are the optimal gestures for the type of action the user must perform within the XR scene.

References

[1] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee *et al.*, "Mediapipe: A framework for building perception pipelines," *arXiv preprint arXiv:1906.08172*, 2019.

[2] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "Mediapipe hands: On-device real-time hand tracking," *arXiv preprint arXiv:2006.10214*, 2020.

[3] R. Egawa and T. Ijiri, *Multi-Window Web Browser with History Tree Visualization for Virtual Reality Environment*. New York, NY, USA: Association for Computing Machinery, 2021, p. 32–34. [Online]. Available: https://doi.org/10.1145/3474349.3480221

[4] T. Waltemate, D. Gall, D. Roth, M. Botsch, and M. E. Latoschik, "The impact of avatar personalization and immersion on virtual body ownership, presence, and emotional response," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 4, pp. 1643–1652, 2018.

[5] F. Argelaguet, L. Hoyet, M. Trico, and A. Lecuyer, "The role of interaction in virtual embodiment: Effects of the virtual hand representation," in *2016 IEEE Virtual Reality (VR)*, 2016, pp. 3–10.

[6] G. Caggianese, N. Capece, U. Erra, L. Gallo, and M. Rinaldi, "Freehand-steering locomotion techniques for immersive virtual environments: A comparative evaluation," *International Journal of Human–Computer Interaction*, vol. 36, no. 18, pp. 1734–1755, 2020. [Online]. Available: https://doi.org/10.1080/10447318.2020.1785151

[7] S. Gupta, S. Bagga, and D. K. Sharma, *Hand Gesture Recognition for Human Computer Interaction and Its Applications in Virtual Reality*. Cham: Springer International Publishing, 2020, pp. 85–105.

[8] K. Kilteni, R. Groten, and M. Slater, "The sense of embodiment in virtual reality," *Presence*, vol. 21, no. 4, pp. 373–387, 2012.

[9] M. Gruosso, N. Capece, and U. Erra, "Exploring Upper Limb Segmentation with Deep Learning for Augmented Virtuality," in *Smart Tools and Apps for Graphics - Eurographics Italian Chapter Conference*, P. Frosini, D. Giorgi, S. Melzi, and E. Rodolà, Eds. The Eurographics Association, 2021.

[10] G. Caggianese, L. Gallo, and P. Neroni, "Design and preliminary evaluation of free-hand travel techniques for wearable immersive virtual reality systems with egocentric sensing," in *International Conference on Augmented and Virtual Reality*. Springer, 2015, pp. 399–408.

[11] B. Fang, F. Sun, H. Liu, and C. Liu, "3d human gesture capturing and recognition by the immu-based data glove," *Neurocomputing*, vol. 277, pp. 198–207, 2018, hierarchical Extreme Learning Machines. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231217314054

[12] X. Huang, Q. Wang, S. Zang, J. Wan, G. Yang, Y. Huang, and X. Ren, "Tracing the motion of finger joints for gesture recognition via sewing rgo-coated fibers onto a textile glove," *IEEE Sensors Journal*, vol. 19, no. 20, pp. 9504–9511, 2019.

[13] H. Ishiyama and S. Kurabayashi, "Monochrome glove: A robust real-time hand gesture recognition method by using a fabric glove with design of structured markers," in *2016 IEEE Virtual Reality (VR)*, 2016, pp. 187–188.

[14] L. Lamberti and F. Camastra, "Real-time hand gesture recognition using a color glove," in *Image Analysis and Processing – ICIAP 2011*, G. Maino and G. L. Foresti, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 365–373.

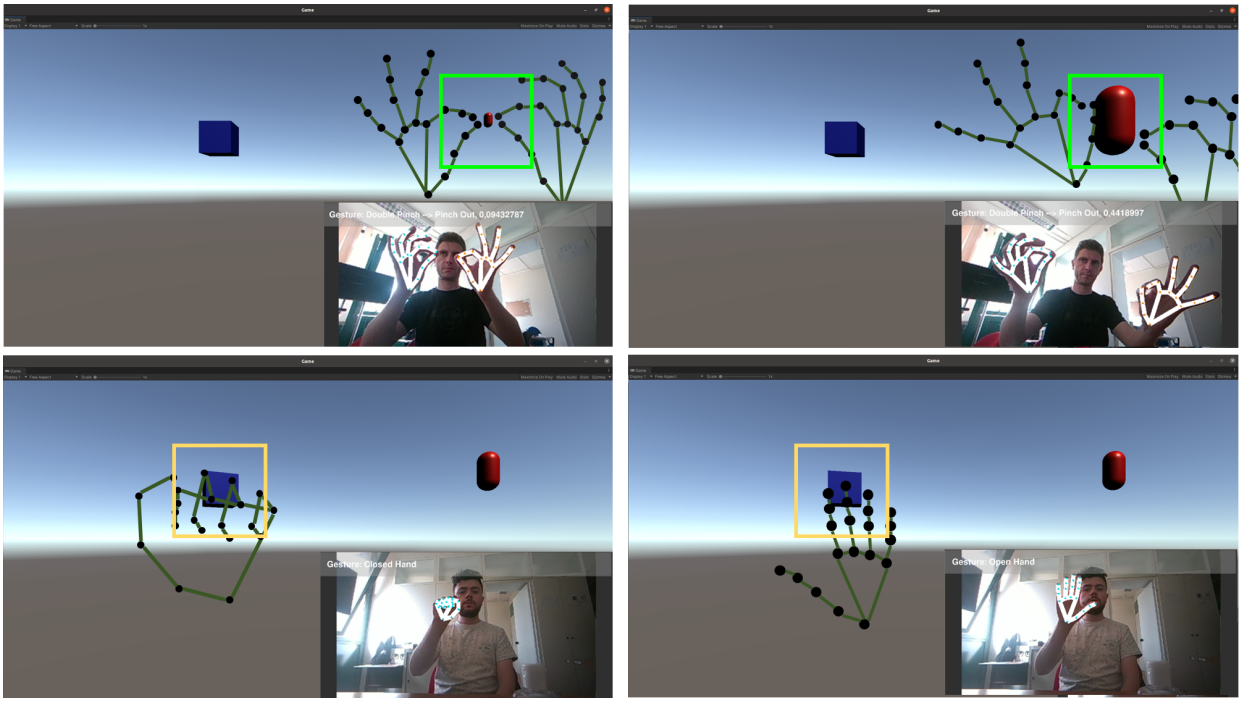

Fig. 5: An example of our system in use. This figure shows how to move objects in a virtual scene using hand gestures and showing their tracking in AR. If necessary, it is possible to view and interact with the elements of the scene in AR.

[15] J. Lin, Y. Wu, and T. Huang, "3d model-based hand tracking using stochastic direct search method," in *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, 2004, pp. 693–698.

[16] J. Segen and S. Kumar, "Gesture vr: Vision-based 3d hand interace for spatial interaction." in *MULTIMEDIA '98*, 01 1998, pp. 455–464.

[17] A. Makris, N. Kyriazis, and A. A. Argyros, "Hierarchical particle filtering for 3d hand tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2015.

[18] S. Sridhar, H. Rhodin, H.-P. Seidel, A. Oulasvirta, and C. Theobalt, "Real-time hand tracking using a sum of anisotropic gaussians model," in *2014 2nd International Conference on 3D Vision*, vol. 1, 2014, pp. 319–326.

[19] F. Mueller, D. Mehta, O. Sotnychenko, S. Sridhar, D. Casas, and C. Theobalt, "Real-time hand tracking under occlusion from an egocentric rgb-d sensor," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[20] F. Yang, Y. Wu, S. Sakti, and S. Nakamura, "Make skeleton-based action recognition model smaller, faster and better," in *Proceedings of the ACM Multimedia Asia*, ser. MMAsia '19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3338533.3366569

[21] A. Caputo, A. Giachetti, S. Soso, D. Pintani, A. D'Eusanio, S. Pini, G. Borghi, A. Simoni, R. Vezzani, R. Cucchiara, A. Ranieri, F. Giannini, K. Lupinetti, M. Monti, M. Maghoumi, J. Jr, M.-Q. Le, H.-D. Nguyen, and M.-T. Tran, "Shrec 2021: Skeleton-based hand gesture recognition in the wild," *Computers & Graphics*, vol. 99, 07 2021.

[22] V. Chheang, P. Saalfeld, F. Joeres, C. Boedecker, T. Huber, F. Huettl, H. Lang, B. Preim, and C. Hansen, "A collaborative virtual reality environment for liver surgery planning," *Computers & Graphics*, vol. 99, pp. 234–246, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0097849321001400

[23] N. Capece, U. Erra, G. Losasso, and F. D'Andria, "Design and implementation of a web-based collaborative authoring tool for the virtual reality," in *2019 15th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*, 2019, pp. 603–610.

[24] T. Piumsomboon, A. Dey, B. Ens, G. Lee, and M. Billinghurst, "[poster] covar: Mixed-platform remote collaborative augmented and virtual realities system with shared collaboration cues," in *2017 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*, 2017, pp. 218–219.

[25] A. Vysockỳ, S. Grushko, P. Oščádal, T. Kot, J. Babjak, R. Jánoš, M. Sukop, and Z. Bobovskỳ, "Analysis of precision and stability of hand tracking with leap motion sensor," *Sensors*, vol. 20, no. 15, p. 4088, 2020.

[26] W. Lu, Z. Tong, and J. Chu, "Dynamic hand gesture recognition with leap motion controller," *IEEE Signal Processing Letters*, vol. 23, no. 9, pp. 1188–1192, 2016.

[27] F. Mueller, F. Bernard, O. Sotnychenko, D. Mehta, S. Sridhar, D. Casas, and C. Theobalt, "Ganerated hands for real-time 3d hand tracking from monocular rgb," in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, June 2018. [Online]. Available: https://handtracker.mpi-inf.mpg.de/projects/GANeratedHands/

[28] M. Gruosso, N. Capece, U. Erra, and F. Angiolillo, "A preliminary investigation into a deep learning implementation for hand tracking on mobile devices," in *2020 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, 2020, pp. 380–385.

[29] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, and M. Grundmann, "Blazeface: Sub-millisecond neural face detection on mobile gpus," *arXiv preprint arXiv:1907.05047*, 2019.

[30] K. Aoyagi, W. Wen, Q. An, S. Hamasaki, H. Yamakawa, Y. Tamura, A. Yamashita, and H. Asama, "Modified sensory feedback enhances the sense of agency during continuous body movements in virtual reality," *Scientific reports*, vol. 11, no. 1, pp. 1–10, 2021.

[31] Q. De Smedt, H. Wannous, J.-P. Vandeborre, J. Guerry, B. Le Saux, and D. Filliat, "Shrec'17 track: 3d hand gesture recognition using a depth and skeletal dataset," in *3DOR-10th Eurographics Workshop on 3D Object Retrieval*, 2017, pp. 1–6.

[32] H. Cheng, L. Yang, and Z. Liu, "Survey on 3d hand gesture recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1659–1673, 2016.

[33] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization." *Journal of machine learning research*, vol. 13, no. 2, 2012.

[34] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[35] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.