Computer Animation and Virtual Worlds

WILEY

RESEARCH ARTICLE OPEN ACCESS

A Multi-User Virtual Reality Game Based on Gesture Recognition From RGB Information

Stefanos Gkoutzios | Michalis Vrigkas

Department of Communication and Digital Media, University of Western Macedonia, Kastoria, Greece

Correspondence: Michalis Vrigkas (mvrigkas@uowm.gr)

Received: 10 December 2024 | **Revised:** 17 February 2025 | **Accepted:** 2 July 2025

Funding: The authors received no specific funding for this work.

Keywords: 3D human-computer interaction | gesture recognition | hand pose tracking | multi-user game | virtual reality

ABSTRACT

In the ever-evolving world of gaming, controller-based input is quickly becoming obsolete. Various applications, including virtual reality (VR) and augmented reality (AR) games, have used motion- and gesture-controlled video game consoles. The current state of the art relies on depth images of the hand that do not utilize color information from the RGB spectrum. In this work, we focus on the development of an interactive VR game that utilizes hand pose recognition from the RGB domain to increase user experience, but also simplify the functionality of the game. To address this challenge, a 3D multi-user VR game themed around a "tennis match" was developed using the Unity engine. We also investigate whether we can estimate the coordinates of colored objects connected to the hand movement of the players and track human gestures to navigate through the game functions in real time using an RGB camera. Statistical analysis showed that the user experience increased concerning engagement and satisfaction using a more natural form of control that allows players to focus on the excitement of the game without worrying about button presses or joystick movements. Such hand pose recognition systems can be implemented to replace the traditional controller-based entry systems used today.

1 | Introduction

Human hand gestures have been proven to be an important factor of human communication, and while hand pose tracking plays a key function in developing human-machine interfaces for virtual environments. Using hand gestures as a substitute for physical switches or remote controls is a typical example [1–3]. Moreover, recent advances in hand gesture recognition in video games have been a topic of increasing interest and yielded great outcomes in the gaming industry over the past years [4, 5]. Gesture recognition has been used in a wide variety of applications, including virtual reality (VR) and augmented reality (AR) games [6], motion-controlled games [7, 8], and gesture-controlled video game consoles [9] to improve user experience and provide personal engagement so that users may interact with computers, smartphones, tablets, or head-mounted displays.

Originally, gesture recognition technology in video games provided only a few limited pre-programmed commands that the users could perform with their fingers. With the development of more sophisticated algorithms, it has become possible to provide users with more accurate recognition of their underlying gestures [10]. In addition, the use of RGB cameras and other tracking sensors has enabled video game consoles and computers to understand players' hand movements, providing more interactive gaming experiences [11].

Abbreviations: AR, augmented reality; VR, virtual reality.

However, many challenges must be overcome before gesture recognition can be widely adopted in VR games to replace physical hand controllers. For example, accurate and real-time hand localization under uncontrolled environments such as background noise and lighting, where hands may occupy less than 10% of the image or be occluded by other objects, or self-occluded, is still a challenging problem [12]. Moreover, the lack of a large-scale data generation model that can offer real-time processing, transferability, standardization, and generalizability in hand gesture recognition for VR systems, while several gestural systems have not yet been benchmarked, is one of the most important challenges to be addressed. To cope with some of these challenges, we incorporated some VR simulations where a human character performed gestural movements in a virtual environment with an RGB camera recording the gestures along with their corresponding annotations. As a result, a large set of gesture examples was generated and used to deploy a deep neural network to recognize the underlying gestures.

As the controller-based input is becoming obsolete due to the rapid development of real-time hand gesture recognition and tracking in VR game environments, it becomes evident that this technology can be seen as a means of controlling gaming experiences. In this work, a multi-user VR game that simulates a tennis match is developed, where the traditional controller-based entry systems used today are replaced by a gesture recognition module that recognizes and tracks the user's gestural movements using RGB information only. This work aims to investigate at what level the gesture mode on the player's hand movements or object recognition based on its color feature can be directly used as a human-machine communication tool between the user and the VR game to replace the traditional controller-based entry systems used today. We aspire that through the analysis of the results, we may be able to understand and determine the factors that constitute a VR game based on gesture recognition appealing to players.

The main contributions of this work can be summarized as follows:

- A hand gesture recognition model is developed that deploys only RGB information from video streams to determine the hand joints' position and estimate the 3D hand keypoints in real-time.
- An object tracker is also deployed that allows for fast and accurate hand tracking based on color information.
- A new multi-user VR game that simulates a tennis match is developed to provide an engaging experience to users. The user's gestures during the game correspond to the hitting of the virtual ball from a racket directly by the physical movements of the player and are mapped to execution commands in the VR environment.
- A suitable evaluation scheme is adopted that evaluates users' experience, satisfaction, and usability. Also, the usability test demonstrates the potential value of substituting the standard physical VR controllers with an integrated gestural recognition control system, which may be engaging and exciting for the users.

The remainder of the paper is organized as follows: In Section 2 a brief review of the related work is presented. In Section 3, the proposed pipeline, including the gesture recognition approach from RGB information, is described, and in Section 4, the implementation details of the VR tennis match game are given. The results of our work concerning the evaluation of the VR game are presented and discussed in Section 5. Finally, the paper summarizes the main findings and draws conclusions in Section 6.

2 | Related Work

2.1 | Hand-Gesture Recognition

Hand gesture recognition systems for virtual reality enhance user interaction by integrating virtual and real-world objects, providing a more immersive experience compared to traditional devices. Several systems employ various image processing algorithms for detection, tracking, and recognition of hand gestures, demonstrating high user acceptability and applicability in virtual reality games [4, 13]. Moreover, hand gesture recognition in VR enables users to interact naturally with virtual environments by interpreting their hand movements and gestures as input commands. Additionally, it facilitates more intuitive communication and interaction, making the VR experience feel more realistic and engaging for users [14].

The most common type of gesture recognition used in video games is known as skeletal tracking [15]. This category of methods uses RGB cameras and depth sensors to capture a 3D image of the player's body and analyzes the images to detect and interpret the corresponding gestures [16]. Thus, gesture recognition can be used to control a variety of game elements, such as character movements, object manipulation, and video game menu navigation. It can also be used to perform special actions, such as spells in role-playing games. For instance, by moving an object or performing specific gestures in racing video games, players may identify with the role of the game character and feel like they are moving naturally in the racing field. This may cause the game to be more attractive to the users and also increase the level of immersion [17].

Recent advances in deep neural networks have enabled computers to recognize and track hand and finger movements easily, faster, and with high accuracy [18]. For example, some video games have implemented gesture recognition algorithms that allow users to perform natural hand movements in a fully immersive environment as they would in the real world [19]. Wang et al. [20] utilized a combination of convolutional neural networks (CNNs) for automatic feature extraction from dual-view RGB images and LightGBM for ensemble learning to perform regression on the extracted features. It employs a dataset acquisition scheme that automatically captures dual-view images and annotates them with corresponding three-axis attitude angle labels using an IMU sensor. This integrated approach allows effective mapping from 2D images to 3D hand attitude angles, addressing challenges such as hand self-occlusion and improving estimation accuracy.

Chen and Huang [21] employed a gesture recognition method that utilizes both a CNN-based approach and the WaveXR plugin

for real-time hand gesture recognition in extended reality environments. The CNN-based gesture recognition model outperformed the WaveXR plugin, achieving high a F1-score indicating that the proposed method provides more accurate gesture recognition compared to the WaveXR plugin's approach. Isabel et al. [22] developed a fully immersive virtual reality system where users can participate in an interactive video game using an HD-sEMG bracelet showing the potential for reliable hand gesture recognition and control applications in VR.

Several promising tools to promote engagement in VR games using personalized hand gesture controllers have been proposed over the last years [23, 24]. For example, [23] proposed a method that involves developing an immersive VR-based hand rehabilitation system that utilizes a personalized gesture-controlled rhythm game, where users perform specific hand gestures to match approaching targets synchronized with music. The authors demonstrated that the proposed VR system successfully activated brain areas associated with motor planning, multisensory integration, and attention, demonstrating its potential effectiveness for enhancing rehabilitation outcomes in healthy individuals and stroke survivors. Moreover, hand gestures in VR systems offer advantages over controllers by reducing fatigue and increasing efficiency, as demonstrated by [24]. Evaluations significantly improved task completion time and user satisfaction, providing valuable insights for future gesture design and enhancing VR interaction techniques.

The combination of virtual reality head-mounted displays and hand gestures used as virtual keyboards is favored for its speed and intuitiveness. To this end, [25] proposed a method that involves a gesture-recognition-based virtual keyboard algorithm designed for head-mounted display devices in augmented and virtual reality environments. Based on the YOLOv3 framework, the proposed method achieves 41 frames per second of real-time processing with high precision. Papadopoulos et al. [26] tried to address the gap in VR gesture datasets by providing two comprehensive datasets. The first dataset refers to controller gestures, while the second dataset refers to hand gestures. These datasets were used to train off-the-shelf time series classifiers to analyze and compare the complexity and performance metrics of hand gesture recognition versus controller gesture recognition with the available online datasets.

However, hand gesture recognition in VR is limited by individual differences in users' hand characteristics and the complexity of gesture deformations, such as variations in hand size and movement styles, which can affect the accuracy of recognition systems. The complexity of gesture movements and the spatiotemporal changes that occur during interactions can also lead to challenges in accurately interpreting gestures. Furthermore, real-time processing requirements and noise in input data, such as hand jittering, complicate the recognition process and may reduce overall recognition performance.

2.2 | Virtual Reality Games

There are many interconnected elements in virtual reality games that work together to create an overall experience [27, 28]. In this context, [29] highlights that the high cost of VR equipment limits its accessibility to a general audience, deterring developers from integrating VR components into games. Also, existing game engines often lack support for necessary VR features, making it hard to incorporate high-end head-mounted displays and other VR technologies. This results in a limited audience for VR systems, posing challenges in creating commercially viable VR games that can reach a broader market.

In their study, [30] proposed a method that involves an experimental setup where participants are divided into two groups, with one experiencing a virtual character with full non-verbal cues and the other with non-verbal cues turned off, to assess the impact on attention and user experience. The scenario includes a preparatory cinematic phase followed by a verbal presentation from the virtual character. Eye-tracking measurements can be used to evaluate shifts in participant focus and engagement throughout the interaction. Vrigkas and Nikou [31] designed and implemented an interactive VR game that utilizes real-time 3D computer graphics to create an object avoidance scenario, allowing players to navigate and interact within a dynamically generated game environment. The game uses mobile devices' accelerometers and compasses to capture orientation and rotation data in 3D space, facilitating user interaction within the virtual environment. This allows the device to display a stereoscopic view of the game, enhancing the immersive experience for players.

A wireless multiplayer interactive VR game transmission framework based on mobile edge computing was proposed by [32]. The game aims to minimize average inter-player delay by optimizing mobile edge computing server resource allocation, wireless bandwidth allocation, and post-processing decision policy, using an iterative algorithm to solve the non-convex problem under various constraints. Kanervisto et al. [33] employed a generative adversarial network (GAN) to create a cheat that mimics human gameplay behavior, enhancing a player's performance in first-person shooters. This approach involves training the GAN on human gameplay data to produce actions that are indistinguishable from those of legitimate players, thereby complicating detection efforts by anti-cheat systems. However, the rise of such cheats may lead to ethical discussions within gaming communities regarding fairness and the overall user experience.

Focused on measuring the user experience in a multi-player VR environment, [34] performed a user study in which participants played two VR games, under varying network conditions, including different access networks and added latency. The authors showed that latency in multiplayer VR gaming had varying effects on user experience depending on the game, with minimal impact from players' prior relationships on social interactions. In this context, [35] performed a content analysis comparing VR-only, VR-supported, and non-VR titles on the Steam digital store, utilizing both automated data-pulling scripts and trained human coders to categorize and analyze various attributes of the applications, including developer categories, user tags, and user ratings. The results proved that VR-only titles received lower positive ratings compared to VR-supported and non-VR titles, highlighting discrepancies in classification systems and user perceptions across different VR experiences.

In recent years, several researchers have studied the effects of VR on user perception, especially on how users perceive and interact

with their environment [36–38]. In the context of the effects of different VR devices on training simulations, [39] focused on user perception and knowledge gain, through two experiments. The first experiment with 61 participants examined the impact of VR displays with varying fields of view on risk detection. The second experiment with 46 participants assessed how different interaction techniques influenced procedural task learning, revealing that users' prior knowledge and gaming experience significantly affect VR simulations and that cybersickness is often due to unawareness of surroundings. Stuart et al. [40] proposed a method that utilizes a web application featuring pre-recorded videos of virtual humans that interact with users via Google DialogFlow, enabling enhanced fidelity in healthcare training simulations. The results indicated that higher fidelity rendering styles are preferred for observing subtle visual cues, while rendering style does not significantly affect interpersonal communication behaviors.

It is important to note that high computational demands and latencies in virtual reality game development mean the extent to which the gameplay will make sense is under threat. In detail, limitations of the hardware concerning the restricted field of view, resolution, and tracking precision have a bearing on user comfort and the degree of immersion achieved. These limitations bear implications for VR game design and usability concerning motion sickness and the requirements for large physical space.

3 | Materials and Methods

This work consists of three modules. The first one is the detection of human gestures in real-time to provide the landmarks of the detected hands and the results of the recognized hand gestures. The second module corresponds to the real-time hand pose tracking module from an RGB camera stream. Finally, the third module is the development of the VR tennis match video game, where the entire gameplay is controlled by the recognized hand gestures. It may be considered as a communication channel between the user's movements in 3D space and the video game. Note that the potential for increased realism is one of the main motivations for replacing traditional hand-held controllers with gestural recognition systems in VR games. This approach can redefine how we play games, allowing for a more intuitive and immersive user experience. Hand gestures are a more natural and intuitive way for users to interact with a virtual environment and feel like being inside a tennis match, holding a racket.

3.1 | Gesture Recognition

In the proposed architecture, we leverage the open-source MediaPipe framework [41] to perform accurate hand pose recognition. More specifically, the proposed framework consists of three gesture recognition steps as follows: (i) a palm detector step that captures RGB images of the hand and rotates the image within an oriented bounding box of the hand, (ii) a hand landmark step that processes the clipped bounding box of the image and returns 3D hand keypoints, and (iii) a gesture recognition step that classifies the 3D keypoints of the hand into a discrete set of gestures.

To determine the different hand positions, we compute 21 hand-knuckle joints that correspond to the image coordinates of the landmarks of the detected hand. Using a 3D joint model, we estimate the joint positions $\mathbf{J} = \{\mathbf{j}_i\}_{i=1}^{J}$ of a hand with $\mathbf{J} \in \mathbb{R}^{3,J}$, where $\mathbf{j}_i = (x_i, y_i, z_i)$ and J is the total number of joints (i.e., 21 joints).

In Figure 1, we demonstrate our hand pose estimation architecture. The RGB information of the palm is given as input to the model, and the VGG-22 network is used to extract a feature map of the palm. In our method, we deploy in all convolutional layers 3×3 kernels with a stride of one and zero-padding of one and 3×3 maxpooling layers. After the last convolutional and fully-connected layers, the ReLU non-linearities are applied. Finally, there are two hidden fully-connected layers with 4096 hidden units each that lead to the output layer that consists of $3, J$ output units. To estimate the 3D joints' position for the output units, a hyperbolic tangent activation function is used.

For the training of the network, we minimized the mean squared error between the estimated 3D joint positions and the ground-truth joints of the network. The loss function $\mathcal{L}$ is given by:

$$\mathcal{L} = \frac{1}{2\mathbf{J}} \|\hat{\mathbf{y}} - \mathbf{y}\|_2^2 = \frac{1}{2\mathbf{J}} \sum_{i=1}^{3\mathbf{J}} \left(\hat{\mathbf{y}}_i - \mathbf{y}_i\right)^2 \tag{1}$$

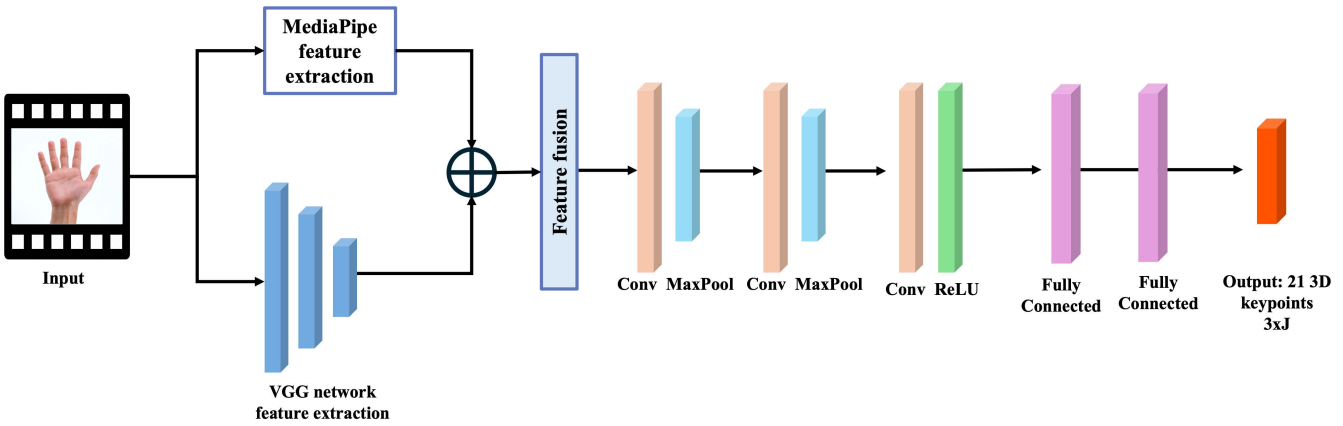
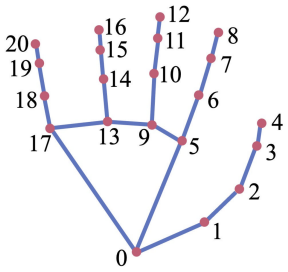


FIGURE 1 | The proposed architecture of hand pose recognition. RGB images are used as input to predict 21 3D keypoints of the hand.



0. Wrist
1. Thumb_cmc
2. Thumb_mcp
3. Thumb_ip
4. Thumb_tip
5. Index_finger_mcp
6. Index_finger_pip
7. Index_finger_dip
8. Index_finger_tip
9. Middle_finger_mcp
10. Middle_finger_pip
11. Middle_finger_dip
12. Middle_finger_tip
13. Ring_finger_mcp
14. Ring_finger_pip
15. Ring_finger_dip
16. Ring_finger_tip
17. Pinky_mcp
18. Pinky_pip
19. Pinky_dip
20. Pinky_tip

FIGURE 2 | The tree-structured hand landmarks with their corresponding indices. Index "0" is the root landmark that represents the hand's base point.

where $\hat{\mathbf{y}} \in \mathbb{R}^{3,J}$ is the 3J-dimensional vector of outputs computed from the network with **J** being the total number of joints, and **y** is the ground-truth vector. To enforce kinematic constraints, we employ the same loss for all joints, since the error for a joint depends on the error for other joints.

Gesture recognition is performed by determining whether each finger of the hand is open or closed. The relative positions of the landmarks to the base point of the palm are depicted as a directed graph tree structure in Figure 2. We may observe that the coordinates of the points $\{4, 8, 12, 16, 20\}$ are the coordinates of the fingertips. The position of the hand with coordinates 0 to 20 is obtained from 21 keypoints, with one coordinate at each joint, at the hand landmark.

To estimate each finger's position, we compare the coordinates on the y-axis of a fingertip with the coordinates of the middle point of the same finger. If the coordinate of a fingertip has a value greater than the coordinate of a middle point, then we set the finger value to one. This means that the index finger is in the open state; otherwise, the finger is closed. For example, to estimate the position of the fingers of one hand, we compare the $\hat{y}_4$ joint with the coordinate of joint $\hat{y}_2$. If $\hat{y}_4 > \hat{y}_2$ then the finger is open, else if $\hat{y}_4 \leq \hat{y}_2$ then the finger is closed.

To cope with lighting variability, the proposed hand-gesture recognition model employs adaptive histogram equalization and brightness normalization. Thus, the contrast of the image is adjusted based on the pixel's neighborhood by dividing the input image into small 9×9 regions and then equalizing the histogram of each region. Moreover, adaptive brightness power-law filtering in the space domain was used to raise the pixel values of the input image to a power. The power-law transformations have the following form:

$$s = c * r^{\gamma} \tag{2}$$

where r corresponds to the pixels of the input image, s is the filtered image, c is a positive constant, and $\gamma > 0$ is a constant that gives a whole family of curves. For values where $\gamma < 1$, the filter maps a narrow range of dark input values into a wider range of output values, while for $\gamma > 1$, the filter reduces the brightness of the input image.e.

Moreover, the background noise is reduced even in complex or dynamic backgrounds since the MediaPipe model is a pre-trained model that provides landmark-based representations of hand gestures rather than relying solely on pixel-level data. This makes the proposed model more robust to background noise. However, if the landmarks are inaccurately detected due to occlusions or overlapping objects, the model may lead to misclassified hand gestures.

The main goal is to recognize the players' gestures from the game interface screen and display a menu of video game functions. The six gestures that are used in the game are shown in Figure 3. The six gestures used to control the VR game environment correspond to (a) select single-player domain, (b) pause game, (c) select multi-player domain, (d) exit game, (e) continue, and (f) return to the main screen.

From the game interface menu, when all fingers are closed, the user may choose to play the game with one player. While with two fingers open, the user may select the multi-user version of the VR game as illustrated in Figure 4. During gameplay, if only one finger is open, it indicates that the game can pause for as long as the user keeps the three fingers to continue the game. Finally, the user has the option either to exit the game (i.e., four open fingers should be recognized) or return to the main screen (i.e., five open fingers should be recognized).

3.2 | Object Tracking

In this section, the approach of real-time hand-pose tracking through color information is analyzed. The proposed implementation uses only the RGB color information, where the colors magenta and green are determined for target tracking from the continuous video stream. These two colors were selected to match the colors of the players in the game and the colors of the racket that the user holds to track his/her movements.

To locate the hand-pose position in the video stream and follow its movement over time, we deployed the YOLOv8 detector, which is fast and effective for real-time applications. A challenging aspect of tracking an object is that it may be occluded by other objects or its appearance may change when different factors, such as lighting, are altered. To address these challenges, we transformed the RGB video stream to the HSV color model since the latter can cope with illumination changes. This color model describes the color in terms of the amount of gray level and its brightness value. Hue extends from 0 to 179, saturation value extends from 0 to 255, and luminance value extends from 0 to 255, respectively. The corresponding mask is created by defining the range of the desired color to be detected (i.e., magenta and green).

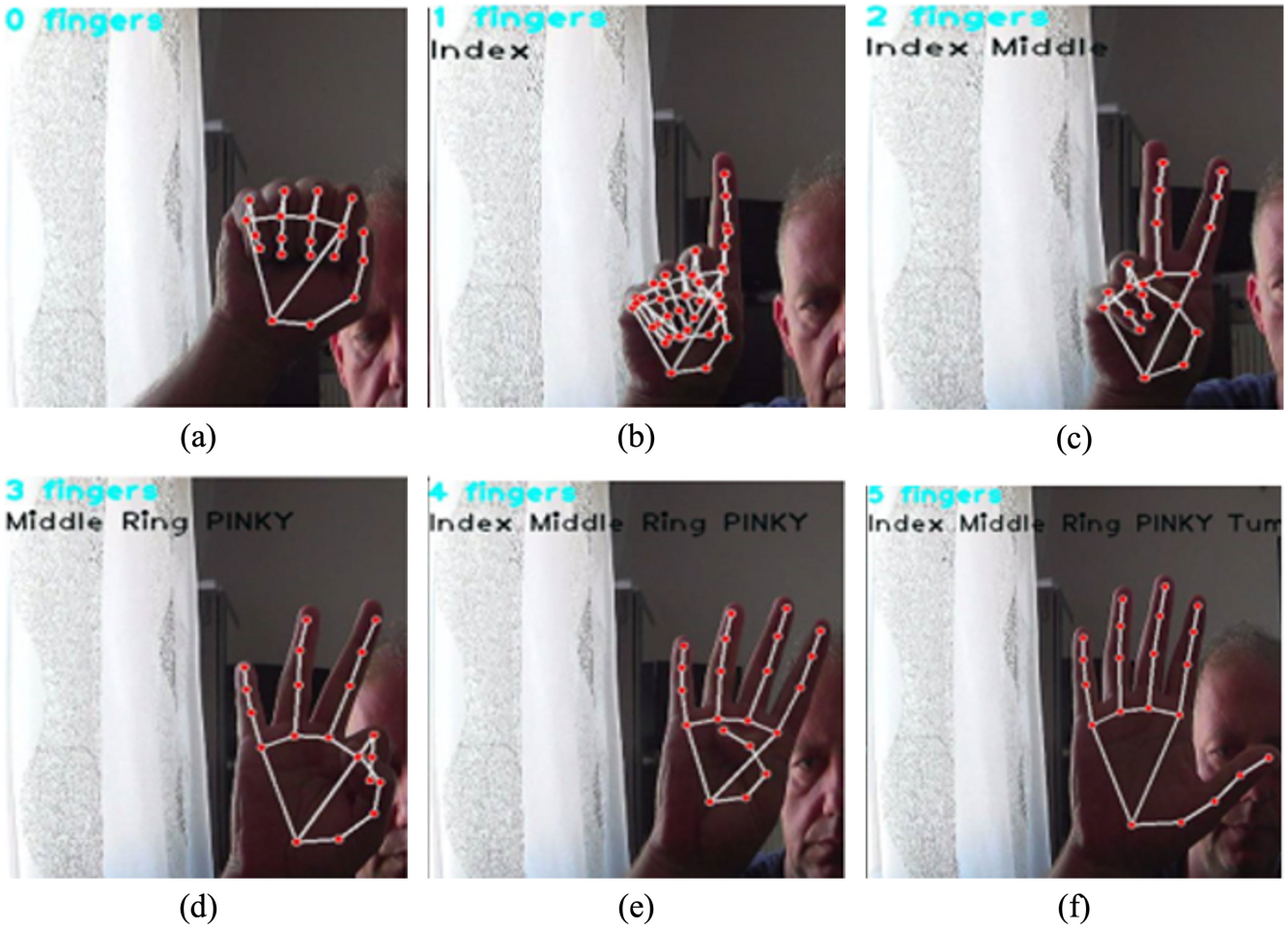


FIGURE 3 | Illustration of the six gestures recognized for operating key game functions. (a) Single-player, (b) pause game, (c) multi-player, (d) continue, (e) exit game, and (f) return to main screen.

Moreover, the YOLO architecture contains 24 convolutional layers and two fully connected layers. An image is divided into a grid of a certain size, and then the network computes for each tile several bounding boxes, assigning to them a confidence score, which indicates how accurate the detected object was concerning the corresponding bounding box. The confidence score is given by:

$$\text{conf} = P(\text{object}|\text{box} = i) * IoU_{\text{pred}}^{\text{truth}} \tag{3}$$

where the term $IoU_{\text{pred}}^{\text{truth}} \in [0, 1]$ is called intersection over union, and it represents the degree of overlap between a predicted frame and the ground truth in each bounding box.

Finally, each bounding box includes four predictions: The center (x, y), the corresponding width (w), and the height (h) of the bounding box. An example of the object detection and tracking of the colored objects with the corresponding bounding boxes using the YOLOv8 detector is depicted in Figure 5.

4 | VR Game Development

The concept of the VR game is a 3D game on the theme of a tennis match. The player is set on a tennis court and plays the game either with the computer as an opponent or with a second player by sharing the game display screen. The Unity 3D cross-platform game engine 2018.4.36f1 was used for the development of the VR game along with the C# language. The VR 3D game is designed to run on desktop devices and head-mounted displays. Apart from the RGB information, a compass, accelerometer, and gyroscope are also used in the application to perform passive device tracking.

An example of the user interface of the VR 3D game is depicted in Figure 6. Area 1 in Figure 6 includes a scene window for viewing and editing the scene's environment. The hierarchy window (area 2 in Figure 6) is located on the left side of the user interface. In the user interface, the hierarchy window is located on the left side (area 2 in Figure 6), which contains all game items in the game scene when it is open. Using the hierarchy window, we may select an object to edit its properties. All selected objects are highlighted in orange and can be translated, rotated, and scaled using the gizmo tool. The asset window (area 3 in Figure 6) corresponds to Unity's file manager, which displays all imported items in the project folder. The folder's structure is shown on the left, while its contents are shown on the right. Finally, the inspector window (area 4 in Figure 6) is used to edit the properties of the selected game elements, materials, and models.

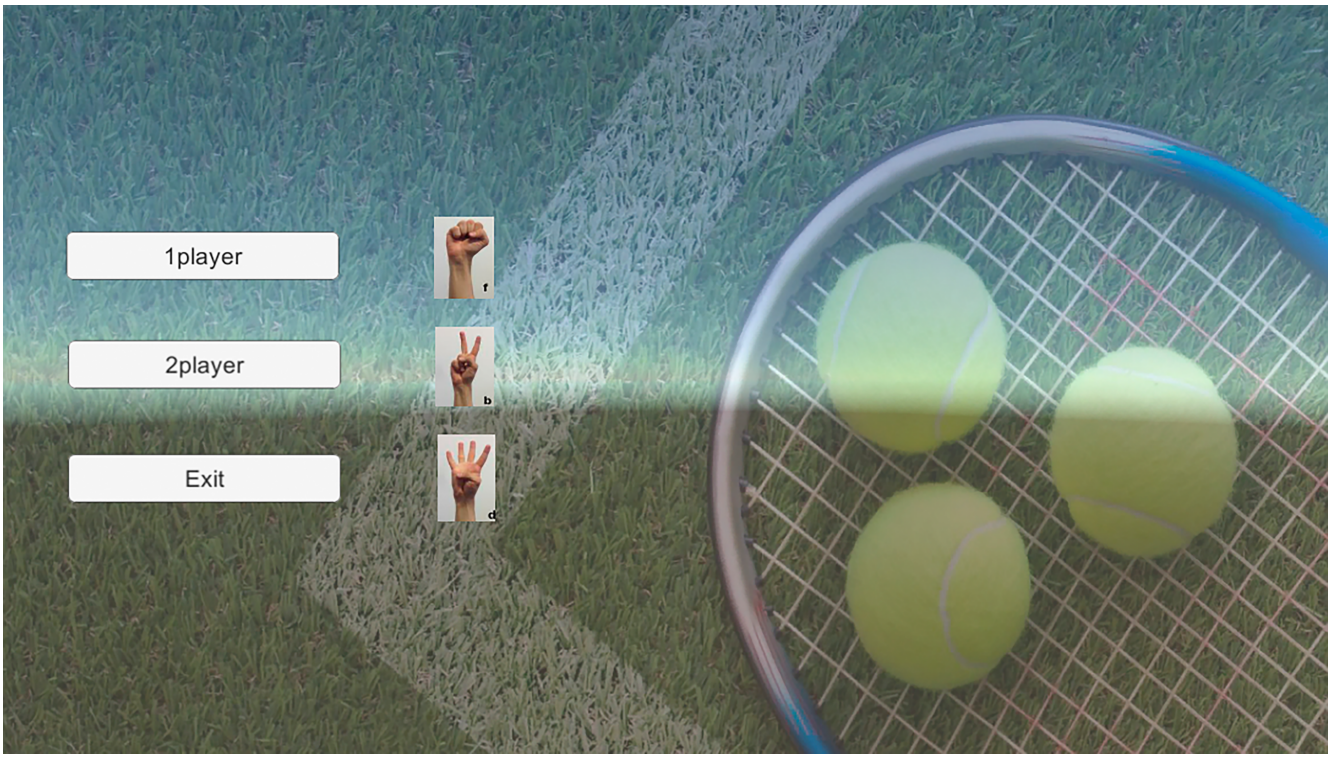


FIGURE 4 | Instance of the game interface menu showing the basic gestures for the operation of the VR game.

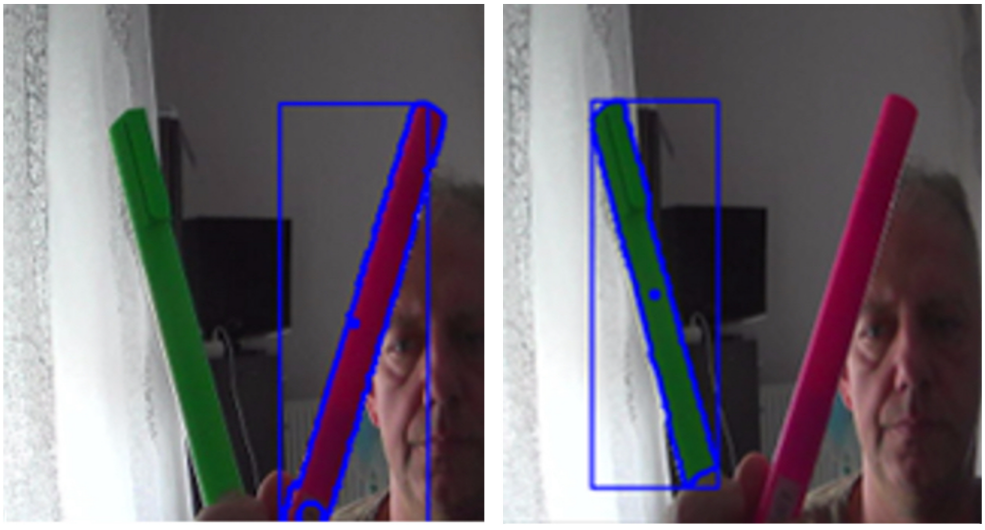


FIGURE 5 | Random instances of real-time object detection and tracking for (a) the magenta and (b) the green object.

Each asset in Unity can be considered as a game object that includes characters, 3D elements, and lighting properties. A game object by itself has no functionality, which means that one must associate several components (e.g., object position and labels, transformation parameters, colliders, and lighting) with it. For example, the default label "Player" is useful for detecting if the player has made contact with the tennis ball, which is performed using the collider control. Figure 7 illustrates an example box collider for a game character of the VR 3D game.

4.1 | VR Game Characters

When players enter the game's virtual environment, they are assigned a 3D character animation as depicted in Figure 8. In the single-user environment, the main character is the *Pink* player who plays against an artificial intelligence (AI) *Bot* player implemented in the Unity game engine. In the case of a multi-user domain, the second player is assigned to the *Green* character. The pink and green colors were selected to correspond with the color characteristics of the objects (i.e., rackets) that the players

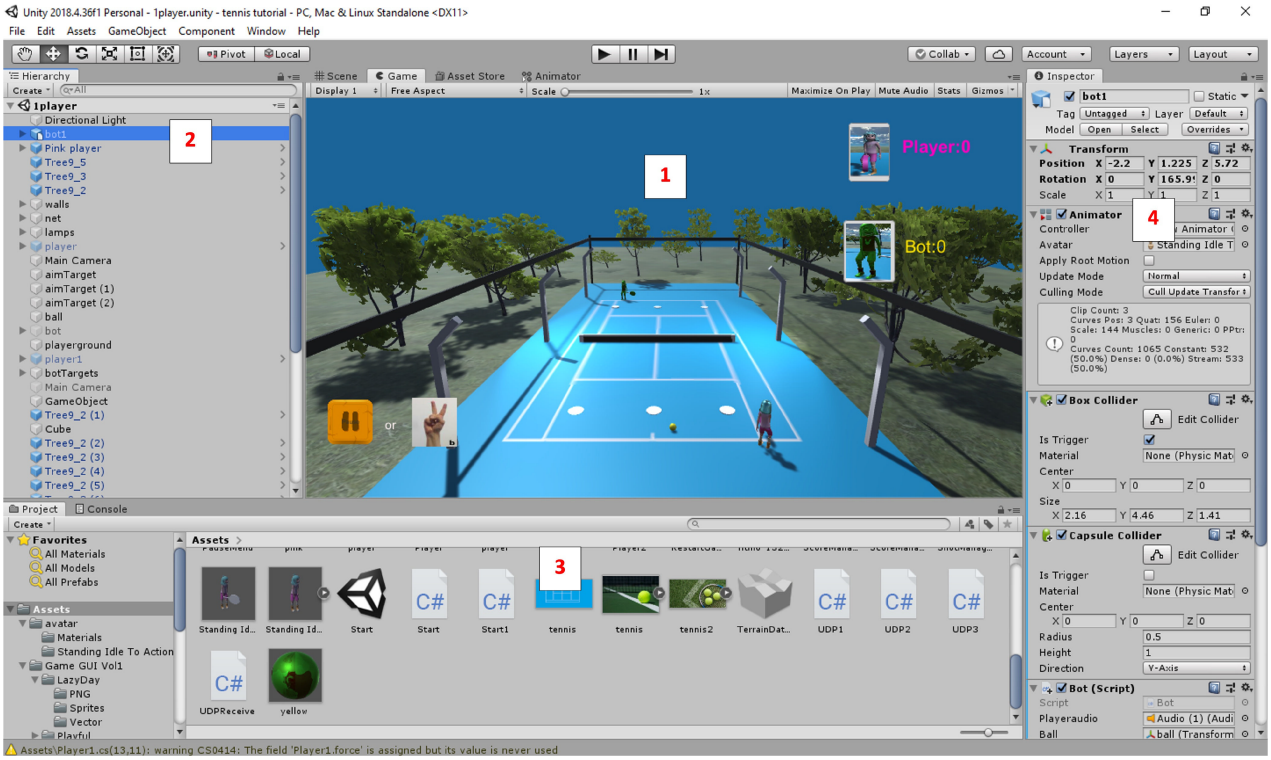


FIGURE 6 | User Interface in Unity game engine.

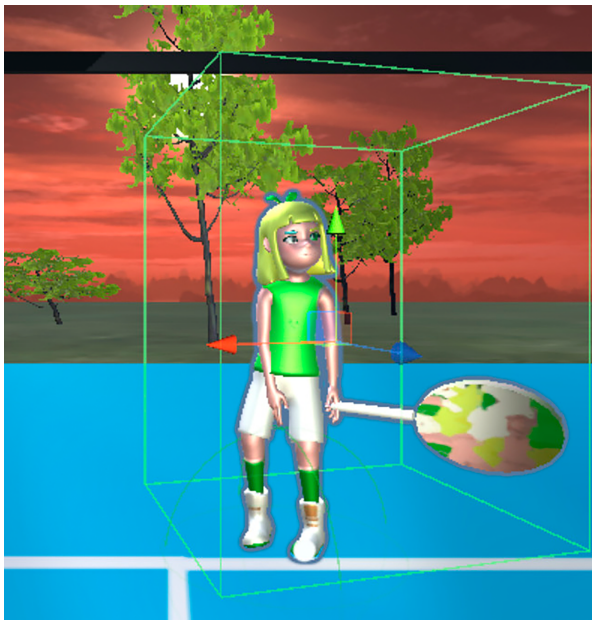


FIGURE 7 | VR game character box collider.

FIGURE 8 | Random instances of the 3D game characters. (a) The *Pink* character, (b) the *Green* character.

should hold in their hands to track their movements to hit the ball with them.

Moreover, an articulated mocap humanoid model (Figure 9) was fit into the 3D character to control the look and feel of each motion and provide realistic character animation when the players hit the ball with the racket they are holding in their hands. Thus, we may record the players' natural movements and translate them into animation data for use in the 3D game.

Finally, Blender 4.0 was used to link the mocap data with the animation rig for the skeleton character.

4.2 | Character Controller

A character controller is a game object that controls a first or third-person character within the Unity game engine. It contains

FIGURE 9 | Example of the standard skeleton overlaid on the character mesh.

all the elements that make up the character of the VR 3D game, such as performance, collision, and transformation parameters (i.e., rotation, translation, and scale). Note that the developed tennis VR game is a third-person game, designed to make the users feel as if they are on the side of the game world, that is, inside a court.

To move a character in the Unity game engine, the character must be scripted and react to user input from an input device (e.g., PC controllers, RGB cameras, or Oculus Quest VR headsets). In our case, the input source comes from RGB cameras, where the objects that the player holds in his hands are recognized and tracked during the gameplay to control the character movements on the set.

4.3 | VR Game Rules

During the gameplay, we keep track of and display the following information according to their goal:

- *Pink player points*: This value corresponds to the points of the first player in each round of the game.
- *Green player points*: If a multi-user environment is selected, this value refers to the points of the second player in each round of the game.
- *Bot player points*: This value corresponds to the points of the *Bot* player.
- *Final score*: The total score of the winner is displayed in this value.

At the start, both players have zero points. To increase the points, the player must hit the ball within the limits of the opponent's area, and the opponent should not repel the ball, or the ball goes outside the boundaries of the tennis court area. The winner is the player with the highest score collected in three different sets of the game. For a player to be declared a winner in each set, he/she has to achieve 15 points. The degree of difficulty of the game also increases within the different sets. The player who wins three sets (i.e., achieves 45 points) is the winner of the game. Some random examples of the gameplay and the different set wins are displayed in Figure 10.

To provide a higher-level view of the system, the use-case diagram of the VR game is depicted in Figure 11. At the beginning of the game, the user has the following three options: (i) start a single-player game, (ii) start a multi-player game, or (iii) exit the game. If the user chooses to play the game, then three cases may occur: (i) play and win the game, (ii) play and lose, or (iii) try the game from the beginning.

4.4 | Lighting Settings

Lighting plays a crucial role in creating realistic graphics in a game and has a dramatic effect on its appearance. By placing appropriate lighting on the scene, we achieve a more realistic result for the player. Lighting in Unity can be divided into two categories: (i) real-time lighting and (ii) rendered lighting. Usually, these techniques are combined to make lighting settings easy to use. Real-time lighting is mainly used to illuminate moving objects such as characters or vehicles. However, because real-time lighting is computed at runtime, it can be very resource-intensive. This is also why real-time light rays do not produce bouncing light.

To create more realistic lighting, global illumination is required. Global illumination is a term used to describe light that reflects or bounces off other objects. However, using global illumination may be too slow to perform in real-time. To avoid this obstacle, a burning lightmap method is used. This method computes global illumination data and stores it in maps called lightmaps. These computations are done during the gameplay. However, a disadvantage of this method is that it may only be used for static objects, since, due to the lightmap, the textures cannot be updated in real-time. This is why most games use a combination of both rendered and real-time lighting. Note that before the rendering, properties such as the number of light bounces computed for the global illumination and the resolution of the lightmap must be specified. These settings affect the duration of the rendering.

4.5 | Communication Protocol

Recall that the operation of the interface menu and the movements of the game's hero are performed exclusively by recognizing and tracking colored objects and the user's gestures. To achieve this, we employ the user datagram protocol (UDP) as the main communication protocol between the received information from the RGB camera and the Unity platform during gameplay. UDP was selected because it offers live media streaming services and is most commonly used in online multiplayer games. This is because UDP sends the packets without waiting for a confirmation before sending the next packet, which means that UDP communication may lead to much lower bandwidth with no time delay.

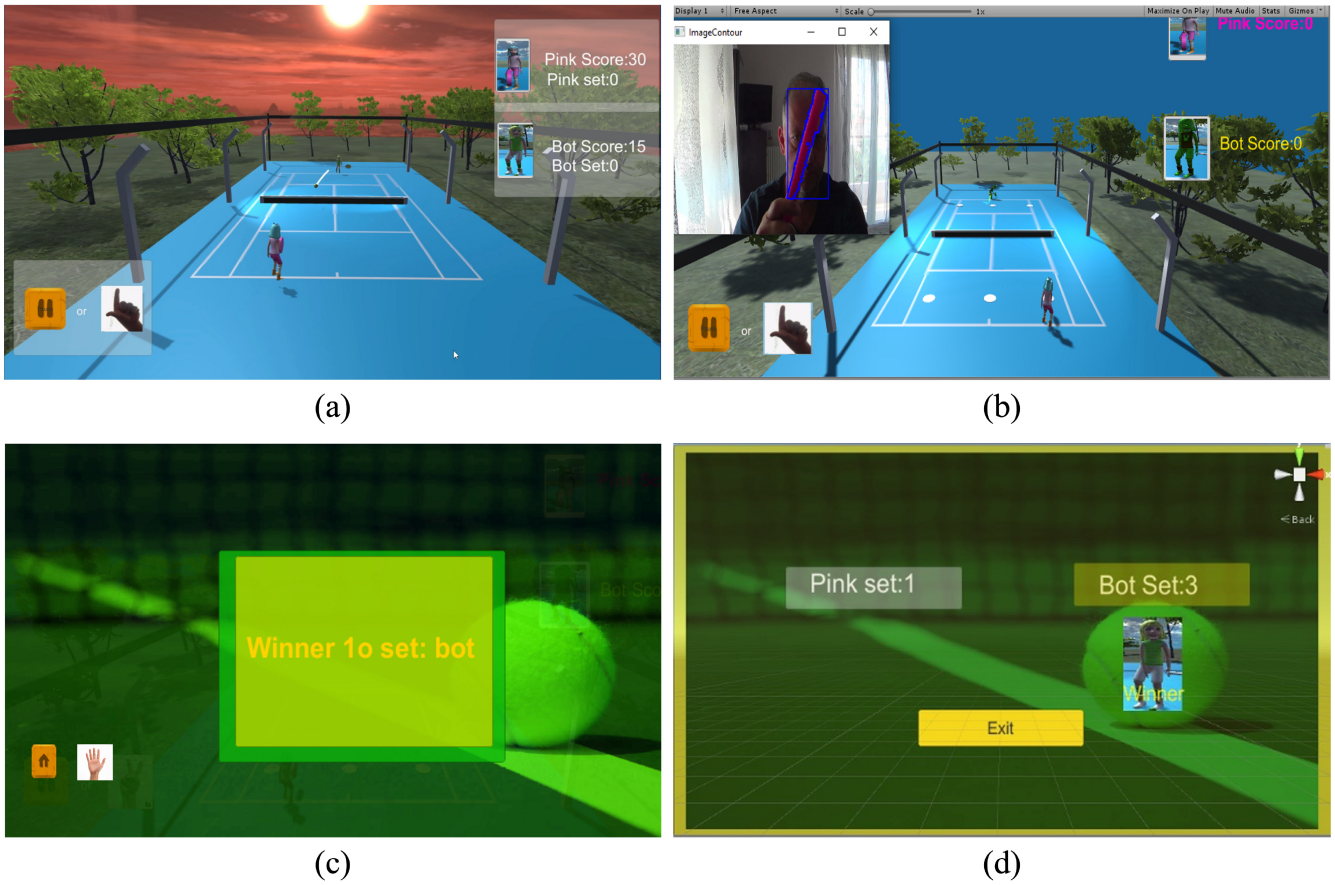


FIGURE 10 | Random instances of the gameplay and the different set wins. (a) Example of the single-player domain where the *Pink* player fights against the *Bot*. (b) Example of the single-player showing the object tracking window. (c) First game set winner screen, and (d) final score and winner of the game.

The socket library is used to send the relevant data via UDP to the following numbered ports:

- *Port 5052*: The identification data of the magenta color is sent.
- *Port 5053*: The identification data of the green color is sent.
- *Port 5054*: The underlying hand gesture recognition data are sent.

The gesture recognition module checks if a hand gesture is given by the user or if a colored object is recognized in the input source. If any of them is detected, then the data from the output of the detection algorithm is sent directly via the UDP protocol to the corresponding Unity objects. The data received by the objects is finally decoded and displayed in the Unity scene.

5 | Evaluation and Discussion

Next, we evaluate the deep neural network architecture for the task of hand gesture recognition. For the evaluation, we employed the same dataset as described by [42]. The dataset contains 2800 sequences of hand skeletons representing 14 classes. The ground truth contains 22 annotated landmarks; however, for comparison purposes with prior work [41], we used only a subset of 21 annotated joint landmarks (i.e., excluding the one that corresponds to the center of the palm). For training the model, we used the Euclidean loss that corresponds to the mean squared error between the ground truth and the predicted landmarks as described in Equation (1). A batch size of 64 training examples per iteration was used. The training was realized for 100 epochs with early stopping, and the Adam optimizer was also used with a learning step of 5×10^{-3} and was decayed by a factor of 0.1. A VR simulation was used to test the model. A human character performed gestures in a virtual environment that were recorded with an RGB camera and their corresponding annotations.

Table 1 compares the performances of different network architectures for the hand-gesture recognition task in terms of F1-score and run-time. The performance of the proposed network achieved an F1-score of 0.931, which emphasizes the effectiveness of our VGG-22 and MediaPipe fusion network architecture. It is also important to note that the recognition is performed in less than 47 ms, which indicates that the proposed scheme may be used for real-time applications.

The hand gesture recognition module consists of a combination of VGG-22 and MediaPipe models. VGG-22 is a deep convolutional neural network that is optimized for real-time performance. It is used for feature extraction by removing its fully

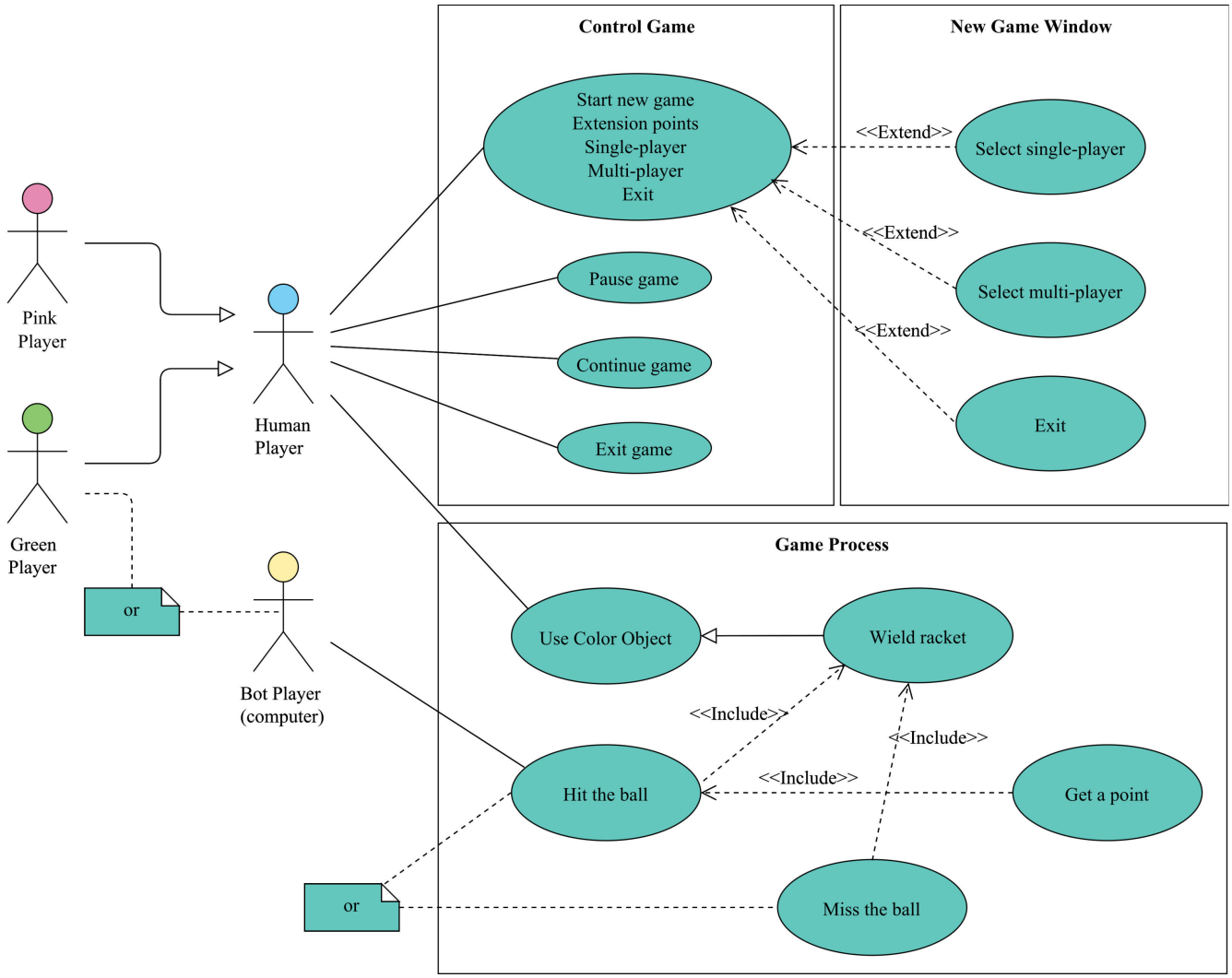


FIGURE 11 | Use-case diagram of the VR game.

TABLE 1 | Evaluation of the hand gesture recognition task.

Model	F1-score	Time (ms)
MediaPipe [41]	0.915	27
VGG-22 [43]	0.921	43
ResNet-50 [44]	0.914	25
Proposed	0.931	47

connected classification layers and keeping the convolutional layers to extract deep spatial features from hand images. MediaPipe provides pre-processed hand landmarks, reducing the computational time for extensive image processing within the network. There were no significant performance trade-offs in the combination of these two models. The entire system was trained using the Titan X GPU hardware acceleration, and an optimization for the batch size was also employed to meet real-time constraints. However, high levels of downsampling and reducing the input image resolution to lower computational time may affect hand gesture recognition. For this reason, a fine-tuning approach to these parameters was conducted to ensure that the model's accuracy remained high without excessive computational overhead.

The corresponding confusion matrices of the hand gesture recognition task are depicted in Figure 12. The proposed methodology, which comprises the combination of VGG-22 and MediaPipe networks, resulted in very small inter- and intra-class classification errors. As it may be observed, only a few classes are confused with each other (e.g., the class "Multi-player" vs. the class "Continue"), while the class "Single-player" was perfectly recognized. Recall that the class "Multi-player" considers two fingers to be open, while the class "Continue" considers that three fingers are open. Since different users may perform the same hand gesture in a different style, it is apparent that the model may appear within class or between different classes variations. To mitigate such ambiguities, a dataset augmentation with multiple users performing the same hand gesture may be implemented to introduce more variations that may boost the proposed model to learn finer differences between similar hand gestures. Furthermore, rotation, scaling, affine transformations, and frame interpolation are implemented to increase the training data size and improve classification accuracy.

Also, fine-tuning the fusion approach between VGG-22 and MediaPipe by giving more weight to key distinguishing landmarks may help the model focus on the most discriminative

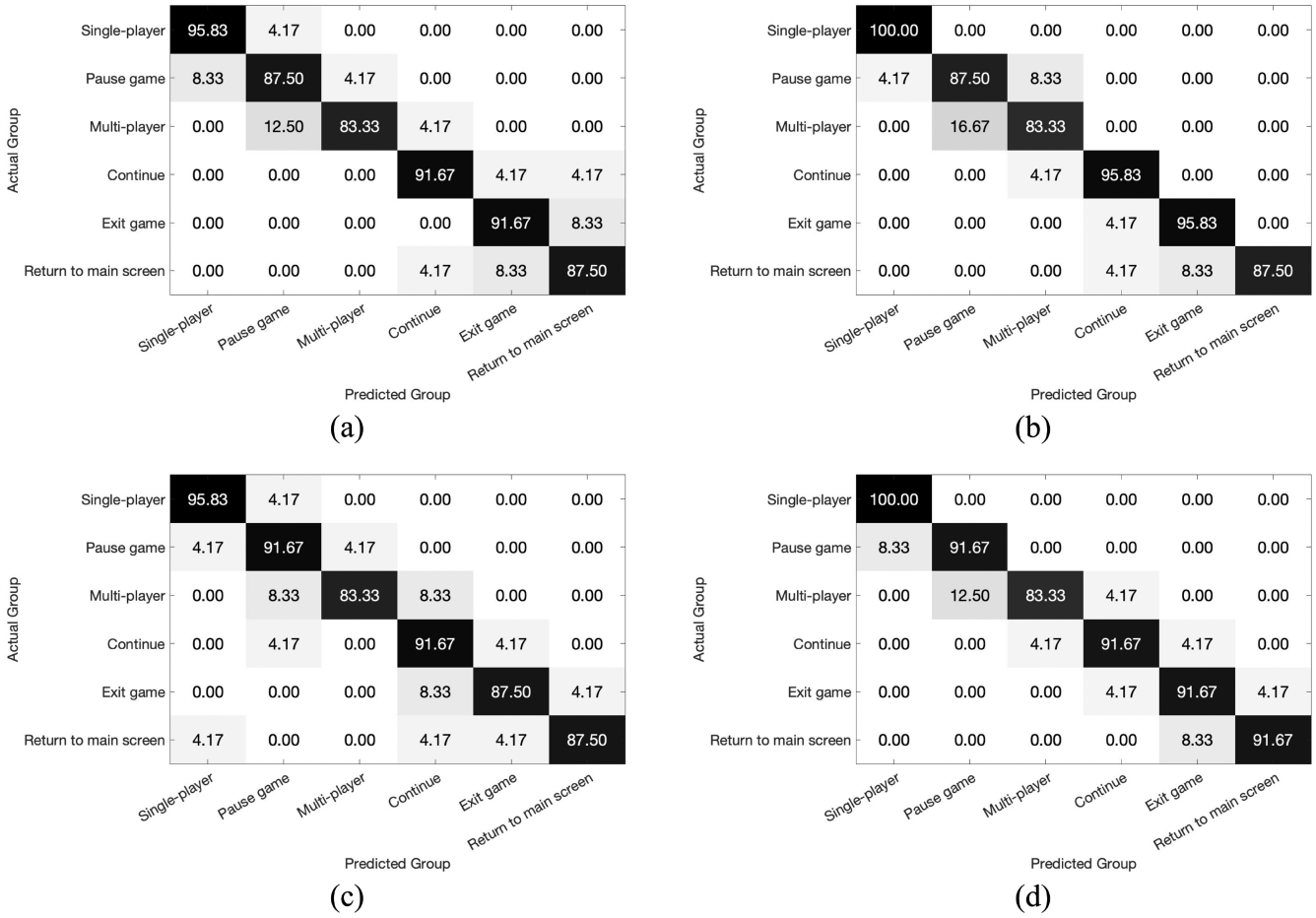


FIGURE 12 | Hand gesture recognition confusion matrices for the (a) MediaPipe, (b) VGG-22, (c) ResNet-50, and (d) proposed methods.

features. Moreover, incorporating temporal information by using sequential models such as long short-term memory (LSTMs) or transformers could provide context and improve hand gesture recognition. However, such models would impose a large computational burden on the proposed model because modeling temporal dependencies requires processing one time step at a time, which can slow down inference, especially in real-time applications.

To evaluate the VR game and analyze the objectives of the study, a self-administered questionnaire of Likert scale items was constructed. The questionnaire was delivered to 52 users (36 male and 16 female, as can be seen in Figure 13) who volunteered for this study, and all of them are actively involved in video game playing. It is also important to note that, in the current study, four participants aged between 42 and 55 also played the tennis match game. However, we excluded these participants because three of them had trouble completing the VR game, while one was not willing to answer the questionnaire. We rated each question on a range of one to five, where one corresponds to strongly disagree and five stands for strongly agree. Participants were asked to complete an anonymous questionnaire after playing the VR game to give their thoughts on the experience, satisfaction, and usability. Additionally, the participants were informed both verbally and in writing that any anonymous data being collected for this study would be accessible to only the researchers of this study, and external access is protected and prohibited.

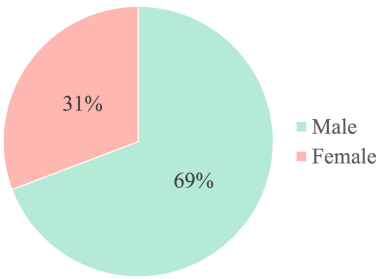


FIGURE 13 | Gender distribution of the participants.

Figure 14 shows the users' experience with the VR game. Based on the results, respondents rated the VR game experience as high and very high regarding their positive feelings from using the game and the level of involvement they experienced.

The scores regarding respondents' satisfaction with the VR game are depicted in Figure 15. Dimensions such as predictions of upcoming events and positive feelings regarding the interaction with the VR environment were measured. The results indicated that high and very high satisfaction scores were exhibited, with more than 75% of the respondents raising positive attitudes from playing the VR game.

Figure 16 represents the users' scores regarding their interaction and the quality of being able to control the events in the VR game.

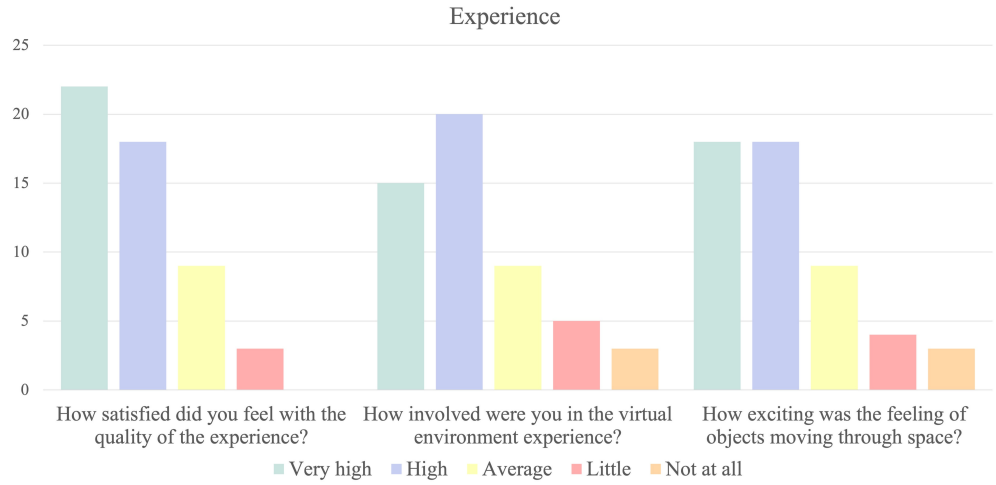


FIGURE 14 | Evaluation of the quality of the users' experience with the VR game.

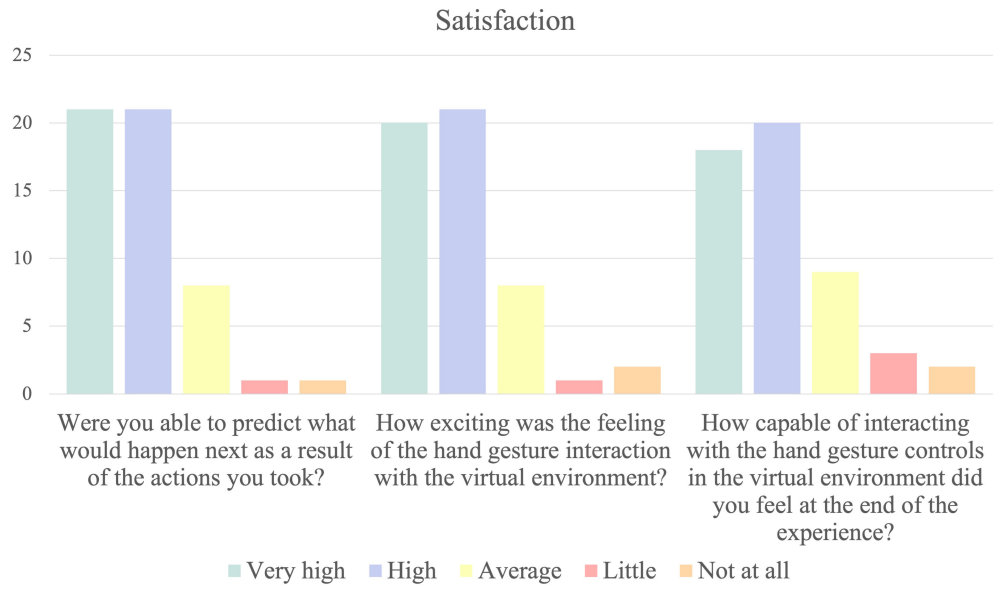


FIGURE 15 | Assessing the user satisfaction with the VR game.

Moderate to high scores were achieved, indicating that the object tracking module provided a good understanding of the VR game rules and a good interaction with the VR game.

The Pearson correlation coefficient was computed to measure the users' satisfaction and usability with the virtual experience of the game. The results are depicted in Table 2. According to the results, the experience dimension is significantly and positively related to the users' satisfaction ($r = 0.752$ and $p = 5.4 \times 10^{-5}$). Similarly, experience with usability ($r = 0.469$ and $p = 0.027$) and experience with immersion ($r = 0.612$ and $p = 0.002$) are also significantly and positively related. Moreover, the satisfaction dimension is positively related to both usability ($r = 0.495$ and $p = 0.019$) and immersion ($r = 0.588$ and $p = 0.006$). Nevertheless, usability did not influence in a significant way with the immersion dimension ($r = 0.335$ and $p = 0.128$). It may be argued that the relationship between usability and immersion is affected greatly by the ability of the users to anticipate the response to their actions and, at the same time, to search and navigate in the virtual environment. Therefore, it is important to consider the level of immersion when designing VR games to actively engage users and enhance their overall experience.

To bridge the gap between usability and immersion, one should consider focusing on enhancing intuitive interaction and, at the same time, increasing user engagement in the VR game. For example, refining gesture-based controls with more natural and personalized interactions allows users to tailor controls based on their comfort level and experience and minimize misclassification errors or delays that could break immersion. Moreover, the proposed VR game incorporates a user tutorial. Also, verbal cues with expert facilitation were given to the users, which may help users master the hand-gesture controls more efficiently, leading to a more fluid and immersive experience.

Furthermore, a between-groups experimental setup was also carried out to facilitate immersive virtual reality and examine the effects on user experience. We split the participants into two

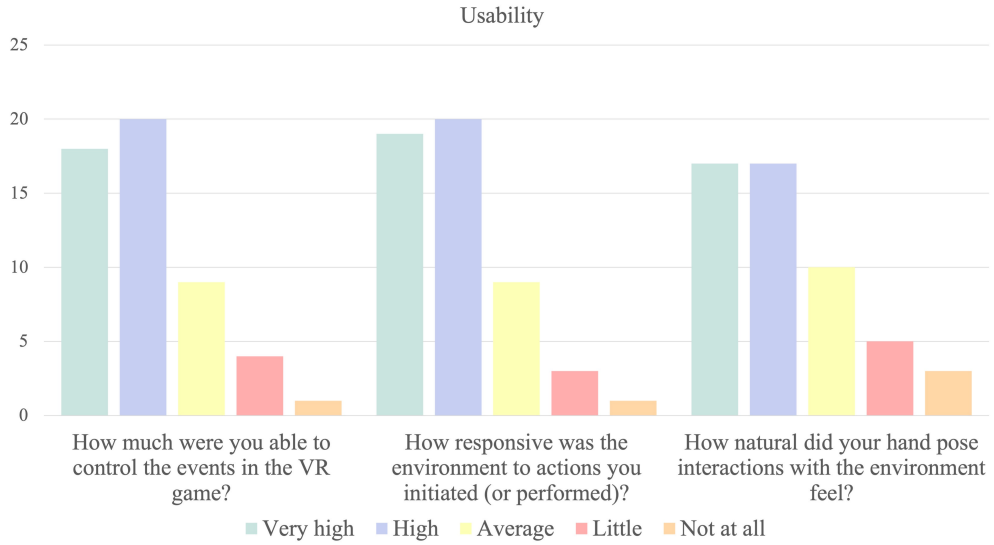


FIGURE 16 | Measuring the usability of the VR game.

TABLE 2 | Pearson correlation analysis.

Relationship	Pearson's r[a]	*P*
Experience–Satisfaction	0.752	5.4×10^{-5}
Experience–Usability	0.469	0.027
Experience–Immersion	0.612	0.002
Satisfaction–Usability	0.495	0.019
Satisfaction–Immersion	0.568	0.006
Usability–Immersion	0.335	0.128

[a]Correlation is significant at the $p < 0.05$ level (two-tailed).

groups, with Group A experiencing the VR game with fewer non-verbal cues and without expert facilitation, and Group B experiencing all the cues in the highest fidelity. From a total of 52 participants, 26 were divided into Group A, and the rest 26 participants were assigned to Group B. Both groups were balanced in terms of gender and age group. Note that the two groups were independent, as each player participated in only one group.

In Table 3, the mean values and standard deviation of the questionnaire items for Group A and Group B are presented. The results from the questionnaire indicate a significant increase in the perception of the VR experience ($p = 5.7 \times 10^{-12}$) and satisfaction ($p = 0.0031$) among participants in Group B as revealed by a paired t-test with a statistical threshold of $p < 0.05$. However, the usability questions (Q7–Q9) did not influence in a significant way ($p = 0.206$).

To ensure that the small sample size in Table 3 does not influence the results and that the observed p values reflect genuine improvements, we calculated Cohen's d to conduct an effect size analysis. The Cohen's d for the perception of the VR experience is 1.34, which is considered to be a large effect size. However, for satisfaction and usability, Cohen's d results are 0.44 and 0.13, respectively. These findings indicate that for satisfaction, Cohen's d is considered to be a medium effect size, while for usability, Cohen's d turns out to be a small effect size.

It is important to note that to measure the usability of the VR game, besides the inquiry method entailing feedback from the users, we also employed a coaching method [45, 46]. To this end, a usability test was carried out after a short presentation of the VR game to the participants. Participants were allowed to ask any system-related questions to an expert facilitator during the usability test. The role of the facilitator was to answer the participants' questions and to steer the user in the right direction while testing the VR game. The purpose of this study is to identify the information users need to provide better training and documentation to them. Thus, participant interaction with the interface can be better understood by analyzing how they use it.

During this test, several concerns arose while using the VR game. Almost 27% of the participants expressed concerns about memorizing the control gestures for operating key functions of the game. To cope with this, a clearer set of instructions has been added to the documentation. We also included the interface of the VR game, relevant icons showing the basic operational gestures as shown in Figures 4 and 10. Moreover, a text message was also included on the initial screen explaining to the users how to start the VR game.

The usability test and the coaching methods showed that the VR game was exciting and user-friendly. Participants were able to operate the VR game using the predefined gesture commands. Also, 64% of the participants indicated that the specific way of handling the game would be interesting if it were applied to other devices such as mobile phones or tablets. Furthermore, more than 70% of the participants reported that they felt freedom of movement during the VR gameplay. Finally, only six participants mentioned that they would expect the game VR character to be able to move forward and back in addition to left-right movement.

Finally, the utilization of the UDP communication protocol for sending data from the RGB camera to the VR game, in practice, proved to be sufficient since no significant time delays were detected nor time lags reported by the users. Thus, the response of the game VR character to the detectable movements of the

TABLE 3 | Evaluation of the questionnaire for the two groups of participants.

		Group A		Group B			
No.	**Questionnaire item**	**Mean**	**SD**	**Mean**	**SD**	**Cohen's d**	**Interpretation**
Q1	How satisfied did you feel with the quality of the experience?	3.42	0.70	4.85	0.37	2.53	Very large effect
Q2	How involved were you in the virtual environment experience?	3.23	1.21	4.27	0.83	1.01	Very large effect
Q3	How exciting was the feeling of objects moving through space?	3.31	1.29	4.38	0.70	1.03	Very large effect
Q4	Were you able to predict what would happen next as a result of the actions you took?	4.00	0.80	4.31	0.97	0.34	Small effect
Q5	How exciting was the feeling of the hand gesture interaction with the virtual environment?	3.69	0.97	4.46	0.86	0.84	Large effect
Q6	How capable of interacting with the hand gesture controls in the virtual environment did you feel at the end of the experience?	3.85	1.05	4.04	1.08	0.18	Small effect
Q7	How much were you able to control the events in the VR game?	3.92	1.06	4.00	0.98	0.08	Very small effect
Q8	How responsive was the environment to actions you initiated (or performed)?	3.73	1.04	4.31	0.84	0.61	Medium effect
Q9	How natural did your hand pose interactions with the environment feel?	3.88	1.21	3.65	1.16	0.19	Small effect

colored object was immediate and without delays. Moreover, the fast recognition of human gestures proved to have a positive impact on the users, resulting in handling the interface screen of the VR game efficiently.

5.1 | Limitations

The game was tested on a standard PC monitor. Perhaps the feel would be more convincing on a larger display screen. Also, a simple RGB camera was used with promising results, possibly only limiting the distance of the user from the camera. To cope with this limitation, the application was also tested with a video camera with greater analysis, and the result was to increase the player's freedom of movement. Also, another limitation is the recognition of user movements only on the X and Y axes. An additional movement of the character along the Z axis may be included in future versions of the VR game to include depth information.

Moreover, the evaluation of the VR game is limited to a small cohort of university students of a certain age (18 to 25 years old) and education level. As a result, the evaluation may be considered age- and education level-biased; thus, it is necessary to evaluate the application features on a larger heterogeneous population cohort to obtain more accurate evaluation results. In this study, addressing the potential bias introduced by excluding older users or individuals with varying technical expertise requires a multi-faceted approach [47]. Younger users between the 18–25 age group may have different cognitive abilities, technology adoption behaviors, and user preferences compared to older individuals. The reason for choosing a younger age group is to test the gesture-controlled tennis match environment between individuals with active involvement in video game playing. Additionally, the experience, expertise, and eagerness of this age group to play video games impose an extra level of difficulty in meeting their satisfaction requirements for a more natural form of game control. Thus, the findings are most relevant to young adults rather than the general population.

6 | Conclusions

In this article, we propose a hand gesture recognition interface that simplifies and streamlines the process of a multi-user VR game. Using hand gestures for communication between the user and the VR game, we explore the possibilities of a fully automated hand gesture control interface in a collaborative workspace. The ultimate goal of this work is to provide a more natural form of control that allows players to focus on the excitement of the tennis match without worrying about button presses or game controller movements. The proposed interface was implemented as part of a virtual reality game engine using RGB sensors, which detect the hands of the users, recognize the underlying hand pose gesture in real-time, and track their movements to hit the virtual object. The system relies on distributed data processing using the UDP protocol to process data from a single RGB camera.

To localize the landmarks of the hands on the RGB video stream from which gestures are recognized, a deep convolutional network based on the VGG-22 and MediaPipe networks was deployed. The position of the landmarks and hand gestures

are used to control the VR game. The combination of MediaPipe and a VGG-22 network is of great importance in hand pose estimation, as the proposed scheme may be insufficient for fusing features extracted from both networks and achieved 93.1% in terms of F1-score on hand pose recognition in real-time. The response of the virtual game character to the detectable movements of the user was in real-time and without delays. The fast detection of human gestures had equally positive results in managing the game's interface screen.

An evaluation study was conducted to measure users' experience and satisfaction with the VR game using a 5-point Likert-scale questionnaire. Also, participants evaluated the features of the VR application during a usability test. Findings indicated that virtual reality games may yield positive feelings of satisfaction with the hand gesture control interface. Furthermore, a high degree of experience for the users was triggered by the VR application. However, this result depends on camera calibration and potentially the operating conditions posed by the ability of the user to adapt to a controller-free virtual environment. In this research, we did not focus on high accuracy, as the developed VR game represents a means for designing a real-time controller-free virtual environment rather than millimeter accuracy, while providing an easy-to-use interface for the user. Despite the subjective nature of the results due to the small cohort used for testing, the results suggest the potential for gestures to increase VR game efficiency.

In addition, future research will focus on improving the hand gesture-based interface using RGB-D sensors, lidar-based sensors, and other types of sensors. A second direction of future development would be to extend the functionalities of the existing VR game with more complex gestures or provide full-body action recognition operations. In addition, augmented reality may be used to implement the visualization system. Also, in future research, to address the potential bias introduced by excluding older users, stratified sampling techniques will be employed to ensure representation from diverse age groups and varying levels of technical expertise.

Author Contributions

Stefanos Gkoutzios: conceptualization, methodology, software, validation, investigation, resources. **Michalis Vrigkas:** conceptualization, validation, formal analysis, resources, data curation, writing – original draft preparation, writing – review and editing, supervision. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

The authors gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research. All statements of fact, opinion, or conclusions contained herein are those of the authors and should not be construed as representing the official views or policies of the sponsors.

Disclosure

The authors have nothing to report.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data presented in this study are available on request from the corresponding author due to privacy or ethical restrictions.

References

1. J. Liu and M. Kavakli, *A Survey of Speech-Hand Gesture Recognition for the Development of Multimodal Interfaces in Computer Games* (IEEE International Conference on Multimedia and Expo, 2010), 1564–1569.

2. X. Guo, W. Xu, W. Q. Tang, and C. Wen, *Research on Optimization of Static Gesture Recognition Based on Convolution Neural Network* (4th International Conference on Mechanical, Control and Computer Engineering, 2019), 398–3982.

3. E. Uboweja, D. Tian, Q. Wang, et al., *On-Device Real-Time Custom Hand Gesture Recognition* (IEEE/CVF International Conference on Computer Vision Workshops, 2023), 4275–4279.

4. S. S. Rautaray and A. Agrawal, *Interaction With Virtual Game Through Hand Gesture Recognition* (International Conference on Multimedia, Signal Processing and Communication Technologies, 2011), 244–247.

5. Y. Zhu and B. Yuan, *Real-Time Hand Gesture Recognition With Kinect for Playing Racing Video Games* (International Joint Conference on Neural Networks, 2014), 3240–3246.

6. S. S. Rani, K. J. Dhrisya, and M. Ahalyadas, *Hand Gesture Control of Virtual Object in Augmented Reality* (International Conference on Advances in Computing, Communications and Informatics, 2017), 1500–1505.

7. B. Xie, H. Liu, R. Alghofaili, et al., "A Review on Virtual Reality Skill Training Applications," *Frontiers in Virtual Reality* 2 (2021): 645153, https://doi.org/10.3389/frvir.2021.645153.

8. C. Cruz-Neira and C. Fernandez, "Virtual Reality and Games," *Multimodal Technologies and Interaction* 2, no. 1 (2018): 1–5, https://doi.org/10.3390/mti2010008.

9. S. S. Oyelere, N. Bouali, R. Kaliisa, G. Obaido, A. A. Yunusa, and E. R. Jimoh, "Exploring the Trends of Educational Virtual Reality Games: A Systematic Review of Empirical Studies," *Smart Learning Environments* 7 (2020): 31, https://doi.org/10.1186/s40561-020-00142-7.

10. J. Galvan-Ruiz, C. M. Travieso-Gonzalez, A. Tejera-Fettmilch, A. Pinan-Roescher, L. Esteban-Hernandez, and L. Dominguez-Quintana, "Perspective and Evolution of Gesture Recognition for Sign Language: A Review," *Sensors* 20, no. 12 (2020): 3571, https://doi.org/10.3390/s20123571.

11. N. V. Le, M. Qarmout, Y. Zhang, H. Zhou, and C. Yang, *Hand Gesture Recognition System for Games* (IEEE Asia-Pacific Conference on Computer Science and Data Engineering, 2021), 1–6.

12. P. Bao, A. I. Maqueda, d. C. R. Blanco, and N. Garcia, "Tiny Hand Gesture Recognition Without Localization via a Deep Convolutional Network," *IEEE Transactions on Consumer Electronics* 63, no. 3 (2017): 251–257, https://doi.org/10.1109/TCE.2017.014971.

13. S. C. Yeh, E. H. K. Wu, Y. R. Lee, R. Vaitheeshwari, and C. W. Chang, "User Experience of Virtual-Reality Interactive Interfaces: A Comparison Between Hand Gesture Recognition and Joystick Control for XRSPACE MANOVA," *Applied Sciences* 12, no. 23 (2022): 12230, https://doi.org/10.3390/app122312230.

14. J. Wang, "Application of Artificial Intelligence in Hand Gesture Recognition with Virtual Reality: Survey and Analysis of Hand Gesture Hardware Selection," 2024.

15. M. A. Livingston, J. Sebastian, Z. Ai, and J. W. Decker, *Performance Measurements for the Microsoft Kinect Skeleton* (IEEE Virtual Reality Workshops, 2012), 119–120.

16. J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced Computer Vision With Microsoft Kinect Sensor: A Review," *IEEE Transactions on*

Cybernetics 43, no. 5 (2013): 1318–1334, https://doi.org/10.1109/TCYB.2013.2265378.

17. S. Xin-Yue, X. Hao-Wei, and W. Mei-Li, *A 3D Hand Joint Detection Network for Real-Time Hand Capture and Its Application in a Game of Moving Mountains* (9th International Conference on Virtual Reality, 2023), 467–473.

18. J. Yu, M. Qin, and S. Z, "Dynamic Gesture Recognition Based on 2D Convolutional Neural Network and Feature Fusion," *Scientific Reports* 12, no. 1 (2022): 1–15, https://doi.org/10.1038/s41598-022-08133-z.

19. M. H. Amir, A. Quek, N. R. B. Sulaiman, and J. See, *DUKE: Enhancing Virtual Reality Based FPS Game With Full-Body Interactions* (13th International Conference on Advances in Computer Entertainment Technology, 2016).

20. P. Ji, X. Wang, F. Ma, J. Feng, and C. Li, "A 3D Hand Attitude Estimation Method for Fixed Hand Posture Based on Dual-View RGB Images," *Sensors* 22, no. 21 (2022): 8410, https://doi.org/10.3390/s22218410.

21. Y. J. Chen and H. S. Huang, "Gesture Recognition Applied to Extended Reality: A Case Study of Online Meeting," in *Icvars '24* (8th International Conference on Virtual and Augmented Reality Simulations, 2024), 84–89.

22. X. Isabel, T. Labbe, F. Chamberland, et al., *Live Demonstration: A Fully Embedded Adaptive Real-Time Hand Gesture Classifier Leveraging HD-sEMG and Deep Learning* (IEEE Biomedical Circuits and Systems Conference, 2023).

23. S. Bae and H. S. Park, "Development of Immersive Virtual Reality-Based Hand Rehabilitation System Using a Gesture-Controlled Rhythm Game With Vibrotactile Feedback: An fNIRS Pilot Study," *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 31 (2023): 3732–3743, https://doi.org/10.1109/TNSRE.2023.3312336.

24. H. Sun, S. Ma, M. Hu, W. Song, and Y. Liu, *Design and Analysis of Interaction Method to Adjust Magnification Function Using Microgestures in VR or AR Applications* (IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops, 2024), 811–812.

25. T. H. Lee, S. Kim, T. Kim, J. S. Kim, and H. J. Lee, "Virtual Keyboards With Real-Time and Robust Deep Learning-Based Gesture Recognition," *IEEE Transactions on Human-Machine Systems* 52, no. 4 (2022): 725–735, https://doi.org/10.1109/THMS.2022.3165165.

26. G. Papadopoulos, A. Doumanoglou, and D. Zarpalas, *VRGestures: Controller and Hand Gesture Datasets for Virtual Reality* (Advances in Computer Graphics: 40th Computer Graphics International Conference, 2024), 336–350.

27. E. Martel, F. Y. Su, J. Gerroir, A. Hassan, A. Girouard, and K. Muldner, *Diving Head-First Into Virtual Reality: Evaluating HMD Control Schemes for VR Games* (International Conference on Foundations of Digital Games, 2015).

28. M. Vrigkas, A. Kleftodimos, and G. Lappas, "An Augmented Reality Workflow for Creating "Live" Wine Labels," *International Journal of Entertainment Technology and Management* 1, no. 4 (2022): 311–327, https://doi.org/10.1504/IJENTTM.2022.10054762.

29. A. Sherstyuk, D. Vincent, and A. Treskunov, "Towards Virtual Reality Games," in *VRCAI '09* (8th International Conference on Virtual Reality Continuum and Its Applications in Industry, 2009), 315–316.

30. E. Dzardanova, V. Nikolakopoulou, V. Kasapakis, S. Vosinakis, I. Xenakis, and D. Gavalas, "Exploring the Impact of Non-Verbal Cues on User Experience in Immersive Virtual Reality," *Computer Animation and Virtual Worlds* 35, no. 1 (2024): e2224, https://doi.org/10.1002/cav.2224.

31. M. Vrigkas and C. Nikou, *A Virtual Reality 3D Game: A Comparison Between an Immersive Virtual Reality Application and a Desktop Experience* (IEEE International Conference on Image Processing Challenges and Workshops (ICIPCW), 2023), 3725–3729.

32. Z. Chen, H. Zhu, L. Song, D. He, and B. Xia, "Wireless Multiplayer Interactive Virtual Reality Game Systems With Edge Computing: Modeling and Optimization," *IEEE Transactions on Wireless Communications* 21, no. 11 (2022): 9684–9699, https://doi.org/10.1109/TWC.2022.3178618.

33. A. Kanervisto, T. Kinnunen, and V. Hautamaki, "GAN-Aimbots: Using Machine Learning for Cheating in First Person Shooters," *IEEE Transactions on Games* 15, no. 4 (2023): 566–579, https://doi.org/10.1109/TG.2022.3173450.

34. S. Vlahovic, I. Slivar, M. Silic, L. Skorin-Kapov, and M. Suznjevic, "Exploring the Facets of the Multiplayer VR Gaming Experience," *ACM Transactions on Multimedia Computing, Communications, and Applications* 20, no. 9 (2024): 1–24, https://doi.org/10.1145/3649897.

35. M. Foxman, D. Beyea, A. P. Leith, R. A. Ratan, V. H. H. Chen, and B. Klebig, "Beyond Genre: Classifying Virtual Reality Experiences," *IEEE Transactions on Games* 14, no. 3 (2022): 466–477, https://doi.org/10.1109/TG.2021.3119521.

36. N. Dollinger, E. Wolf, D. Mal, et al., *Virtual Reality for Mind and Body: Does the Sense of Embodiment Towards a Virtual Body Affect Physical Body Awareness?* (Conference on Human Factors in Computing Systems Extended Abstracts, 2022).

37. B. Schone, J. Kisker, L. Lange, T. Gruber, S. Sylvester, and R. Osinsky, "The Reality of Virtual Reality," *Frontiers in Psychology* 14 (2023): 14, https://doi.org/10.3389/fpsyg.2023.1093014.

38. S. Beata, "Impact of Virtual Reality Cognitive and Motor Exercises on Brain Health," *International Journal of Environmental Research and Public Health* 20, no. 5 (2023): 4150, https://doi.org/10.3390/ijerph20054150.

39. A. Menin, R. Torchelsen, and L. Nedel, "The Effects of VR in Training Simulators: Exploring Perception and Knowledge Gain," *Computers & Graphics* 102 (2022): 402–412, https://doi.org/10.1016/j.cag.2021.09.015.

40. J. Stuart, K. Aul, A. Stephen, M. D. Bumbach, and B. Lok, "The Effect of Virtual Human Rendering Style on User Perceptions of Visual Cues," *Frontiers in Virtual Reality* 3 (2022): 864676, https://doi.org/10.3389/frvir.2022.864676.

41. C. Lugaresi, J. Tang, and H. Nash, "MediaPipe: A Framework for Building Perception Pipelines," 2019.

42. Q. De Smedt, H. Wannous, and J. P. Vandeborre, *Skeleton-Based Dynamic Hand Gesture Recognition* (IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2016), 1206–1214.

43. K. Simonyan and A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition* (3rd International Conference on Learning Representations, 2015), 730–734.

44. K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Learning for Image Recognition* (IEEE Conference on Computer Vision and Pattern Recognition, 2016), 770–778.

45. H. R. Hartson, T. S. Andre, and R. C. Williges, "Criteria for Evaluating Usability Evaluation Methods," *International Journal of Human-Computer Interaction* 15, no. 1 (2003): 145–181.

46. A. Fernandez, E. Insfran, and S. Abrahao, "Usability Evaluation Methods for the Web: A Systematic Mapping Study," *Information and Software Technology* 53, no. 8 (2011): 789–817, https://doi.org/10.1016/j.infsof.2011.02.007.

47. C. H. Chu, R. Nyrup, K. Leslie, et al., "Digital Ageism: Challenges and Opportunities in Artificial Intelligence for Older Adults," *Gerontologist* 62, no. 7 (2022): 947–955, https://doi.org/10.1093/geront/gnab167.