Gesture Play: Revolutionizing Gaming with Computer Vision and Machine Learning

Abhay Chopde
Department of Electronics and Telecommunication Engineering
Vishwakarma Institute of technologyPune, India
abhay.chopde@vit.edu

Balganesh Thombre
Department of Electronics and Telecommunication Engineering
Vishwakarma Institute of technologyPune, India
balganesh.thombre21@vit.edu

Vedant Khodake
Department of Electronics and Telecommunication Engineering
Vishwakarma Institute of Technology,Pune, India
anil.vedant22@vit.edu

Nikhil Yeware
Department of Electronics and Telecommunication Engineering
Vishwakarma Institute of technologyPune, India
nikhil.yeware21@vit.edu

***Abstract—* The project presents a novel angle to control games through virtual controls, substituting real-life physical activity into the system with the power of computer vision and deep learning techniques. This system incorporates integration from MediaPipe's Pose model, OpenCV, and PyAutoGUI for users to proficiently play games by using movements such as running, jumping, and shooting that are translated to keyboard inputs. The MediaPipe Pose model detects body landmarks very accurately in real-time, and video processing of the stream by OpenCV is efficient enough to keep the speed at an average 20 FPS. Critical actions are detected using algorithms custom-made for specific landmark positional data and movement patterns. This state of an application ensures enough visual feedback to the user for meaningful interactivity and seamless state management to prevent erratic input while gaming. This project shows an innovative application of pose estimation and real-time action recognition that will form the basis for many more immersive and natural human-computer interaction experiences in gaming and beyond. Future developments might include gesture customization and multi-user support, expanding its scope of usage.**

Keywords: Visitor verification, alarm feature, real time weapon detection, structural similarity index, LBPH algorithm, python GUI application, crime prevention, data analytics.

I. Introduction

Creating games and interactive systems has increased the requirements for an environment where interaction with digital environments is more immersive and intuitive. Sufficient traditional input devices include keyboards, mouse, and controllers, but these introduce a barrier between the user and the application, thereby limiting natural engagement. This project fills the gap in existing technologies with the development of a real-time game control system that uses the body as inputs to dissociate from traditional hardware to reinforce user immersion. The system is implemented using state-of-the-art computer vision along with deep learning techniques, such as MediaPipe Pose model, OpenCV, and PyAutoGUI for real-time action detection and game control. A MediaPipe Pose model tracks body landmarks such as head, shoulders, and hips. Thus the specific actions like running, jumping, shooting can be identified. OpenCV handles video acquisition and processing in the attached stream of frames, keeping a stable 20 frames per second for a smoother user experience. The detected movement translated through keyboard inputs using PyAutoGUI can handle other sensitive interfaces like games easily. This project shows very practical solutions to pose estimation and real-time video processing in the task of human-computer interaction. It uses are outside gaming systems and can be used for fitness training, virtual reality scenes creation, or gesture-based control systems. It ultimately ends with the extra hardware dependency that denotes an approach that is accessible yet both cost-effective and innovative for creating immersive user interfaces.

II. Literature Review

The proposed research on vision-based hand gesture recognition frameworks, focusing on the use of machine learning algorithms to recognize gestures and posture in real-time within human-computer interaction systems, has reached an accuracy level of 99.4% hand posture recognition and 93.72% dynamic hand gesture recognition. Validation Two applications: (1) Referee Command Language Interface System (ReCLIS), a robotic soccer referee commands system, and (2) a Portuguese Sign Language Recognition system for vowels. The proposed approach focuses on making it very efficient in terms of computation, robust to environmental conditions, and scalable, thus being applicable in very different contexts, including robotics, virtual reality, gaming, and assistive technologies. Future developments involve methods towards stereo vision, continuous gesture recognition and reinforcement learning for user adaptation [1].

Such a game-based rehabilitation system was designed and developed by incorporating hand gesture recognition as a human-computer interaction technique. The system offers real-time control for the virtual mouse, with rehabilitation done through interactive games, enabling increased patient engagement and improved recovery outcomes. Some of the key techniques applied in the process involved RGB based hand segmentation, centroid tracking, and convex-hull-based finger counting to play mouse functions. Actually, the results obtained from the experiments show how the system will work in real-time applications. Therefore, it makes it possible to improve the existing rehabilitation methods by being used for interactivity and fun practice. However, some of the issues like light and background effects were also found in the results that are to be optimized so that they can get robustly deployed in different environments [2].

Recent studies include gesture-based controls with the aim of improving the overall experience of video games. One experiment explored intuitive hand gestures assigned to the game commands and assessed user satisfaction by using the User Experience Questionnaire (UEQ) alongside the gameplay analysis of three mini-games. The outcome was successful in gesture-based interactions at high levels of user satisfaction, being in the top 1% compared to other interactive products from the UEQ database. However, it was inconsistent with respect to gesture reliability in the analysis, which can bring about scopes of improvement in those fields. These results expose the great scope of gesture-based interfaces for the betterment of engagement and interaction in video gaming [3].

This research introduces a low-cost vision system for real-time, automatic hand gesture recognition to enhance the interaction in gaming scenarios. Hand gestures are captured and interpreted by a low-cost web camera and the hull-point analysis algorithm, without using physical input devices such as keyboards or joysticks. A prototype is developed and tested for three interactive games with effective gesture recognition and interactive response. The outputs have indicated that the system could be a powerful tool for game development in immersive games but problems such as lighting conditions and complex gestures will have to be tackled. This research establishes a basis of integrating natural hand gesture with human-computer interaction [4].

This multi modal interaction framework proposed combines 3D gesture tracking with physical game controllers. The system is achieved with a custom 3D depth camera and, in combination with accelerometer and gyroscope data, guarantees accurate six-degree-of-freedom control and effortlessly intertwines gestures with controller movements to offer great immersion and accuracy within virtual environments. Two prototype games, the laser sword fighting game and the third-person action game, demonstrate and illustrate the capabilities of the system, which, by combining the elements of tactile feedback and immersive gestures, can redefine natural interaction in games [5].

III. METHODOLOGY

The innovative use of controlling games is undertaken by converting the movement of the human body to keyboard inputs to allow the real-life experience in the game. The methodology includes four steps-major ones, namely video capture in real time, pose detection, movement analysis, and key mapping to control the game.

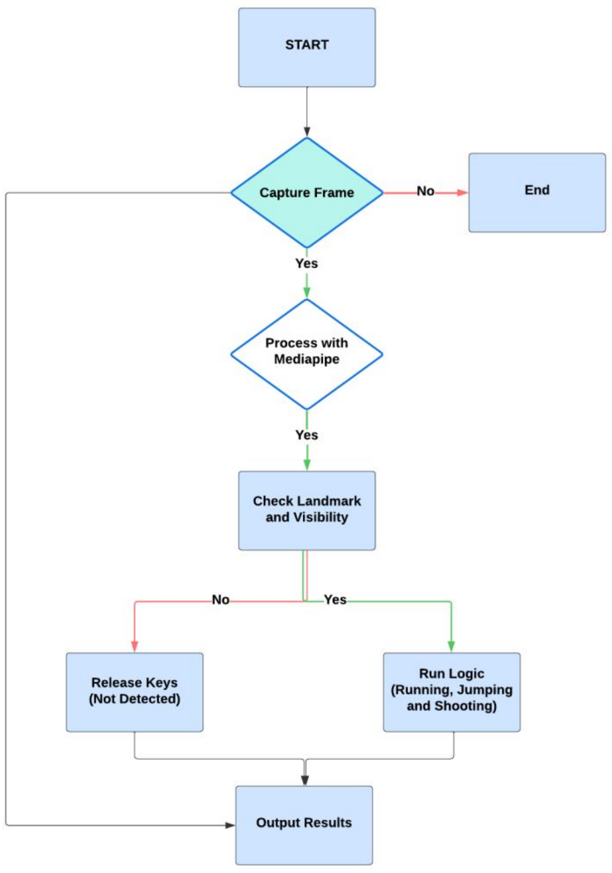
START
Capture Frame
No
End
Yes
Process with Mediapipe
Yes
Check Landmark and Visibility
No
Yes
Release Keys (Not Detected)
Run Logic (Running, Jumping and Shooting)
Output Results

Fig.1. Game Logic Flowchart

i. Real-Time Video Capture

The system begins by capturing a live feed of video from the user through a webcam that is being managed by OpenCV. Then, a VideoCapture object is instantiated in order to capture frames from the camera. Every frame is in real time; all performance issues with smoothness are eliminated so that the frame rate is always consistent at 20 FPS. The video feed is similarly operated in mirror mode so that the movement thus exhibited in the video is intuitive and as expected by the user.

ii. Pose Detection Using MediaPipe

MediaPipe Pose model is employed to detect and track key body landmarks in each frame. This pre-trained model identifies critical points such as shoulders, hips, knees, and ankles, ensuring robust pose estimation under varying conditions. The confidence thresholds for detection and tracking are optimized to maintain accuracy while processing at high speeds. Detected landmarks are then used to extract positional data for movement analysis.

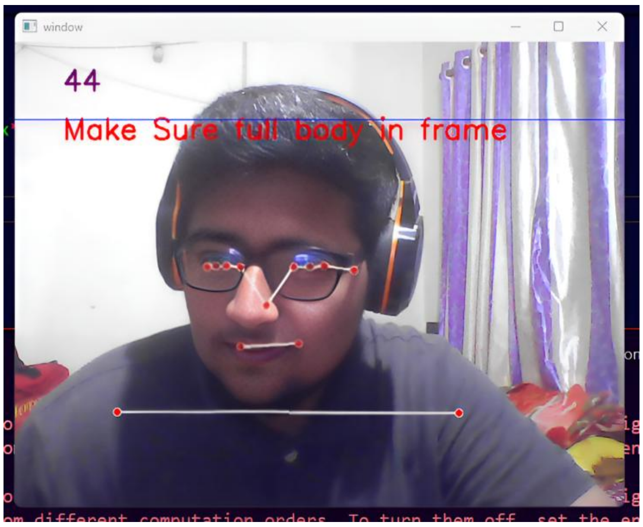

Fig.2. Capturing Frame

This image represents the first step of the system: acquiring a video frame from the webcam. It shows the raw feed from the camera with no processing applied yet. The frame serves as the input for the subsequent pose detection and action recognition stages. Visual indicators such as a bounding rectangle or overlay text might confirm the system is actively capturing video.

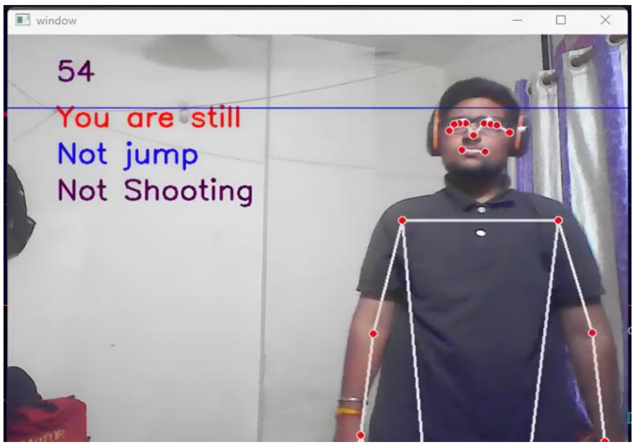

Fig.3. Body Pose Detection

This image demonstrates the successful detection of the user's full body pose. MediaPipe Pose landmarks are overlaid on the live video feed, marking key points like shoulders, hips, knees, and ankles. The pose connections form a skeleton-like representation of the body, verifying that the user's entire frame is within the camera's field of view and landmarks are tracked accurately.

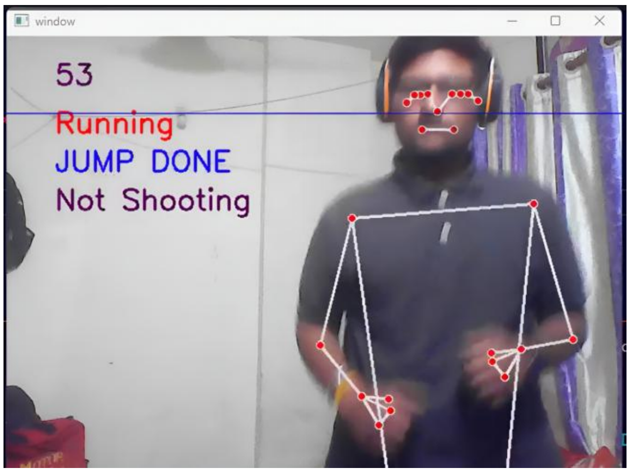

Fig.4. Jumping Pose Detection

This output highlights the system detecting a running motion. The image shows landmarks that suggest coordinated leg and arm movements characteristic of running. A text overlay, such as "Running Detected," is displayed on the frame. Additionally, the "d" key might be simulated and indicated as being pressed.

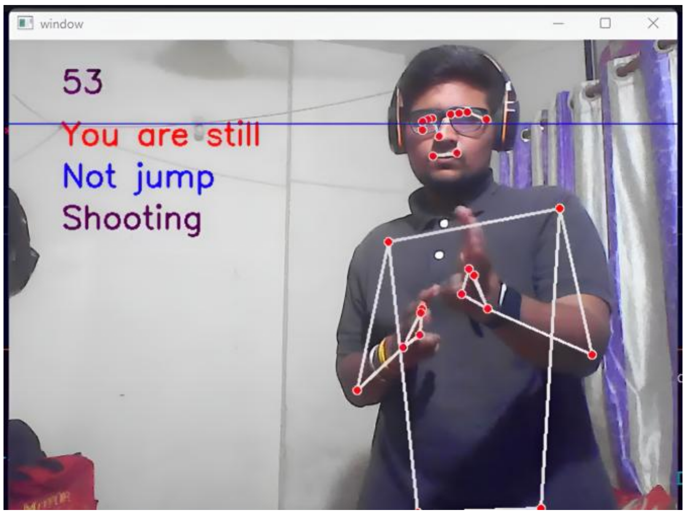

Fig.5. Shooting Pose Detection

In this image, the system identifies the user performing a shooting action. The arm positions and proximity between key landmarks indicate the action. A visual cue, such as "Shooting Detected," is shown on the screen, with the corresponding "s" key simulated in the game. The system ensures precision by checking the relative position of hands and arms.

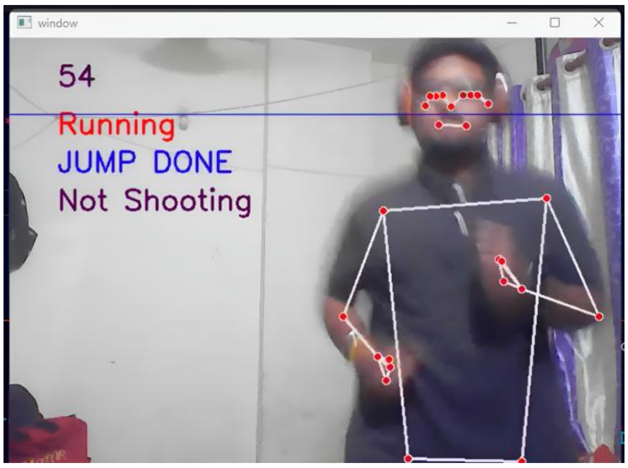

Fig.6. Running Pose Detection

This frame shows the system recognizing a jump. The head's vertical displacement and coordinated body movement suggest an upward motion. The overlay reads "Jump Detected," and the "j" key is virtually pressed. This detection helps confirm that the system accurately responds to quick and dynamic actions.

iii. Movement Analysis

It tracks user actions based on the positional data of detected landmarks in consecutive frames. A jumping motion infers the relative vertical displacement of the head, while coordinated leg movements inform about running. The system distinguishes between different actions like jumping, running, and shooting by making checks against predefined thresholds about the pattern of movement.

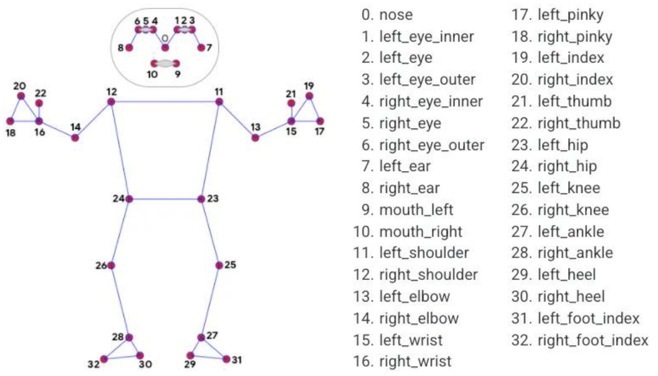

Fig.7. MediaPipe pose detection

iv. Key Mapping and Interaction

It maps the identified actions to appropriate keyboard inputs through PyAutoGUI, for instance, jumping to key "j," and running to key "d" and shooting keys "s." The system also ensures that every key is released automatically at the cessation of action but in synchronization with the action detected. This aids smooth interaction with the game.

v. Feedback Mechanism

To enhance usability, the system overlays real-time visual feedback on the video feed using OpenCV. Indicators such as text annotations and colored markers inform the user about detected actions, ensuring transparency and reducing input errors.

IV. TECHNOLOGY USED

i. MediaPipe Pose Model

Real-time, high-accuracy detection of 33 key body landmarks is achieved through the MediaPipe Pose module, making it good for detecting more complex actions such as running or jumping. The architecture is lightweight enough to be efficiently processed and doesn't require high end hardware in order to execute it.

ii. OpenCV

OpenCV took care of video capture streams via the webcam along with any required frame conversions, mirroring, and overlaying of visual feedback. Easy integration and a very wide library of functions on image processing make OpenCV an inescapable tool for any real-time application.

iii. PyAutoGUI

PyAutoGUI lets a user play games by simulating keypresses and releases seamlessly, based on activity detected. It allows very straightforward syntax and thus makes a great script to translate movements to dynamic keyboard inputs.

iv. Python Programming Language

All of this is stuck together with Python glue that binds all other elements, giving a very flexible and developer-friendly environment. Python has the widest ecosystem of libraries for efficient implementation of anything from real-time video processing to GUI automation.

V. RESULT AND DISCUSSION

The code implements a pose-based control system for the game successfully using computer vision and machine learning approaches. It uses MediaPipe for pose estimation and PyAutoGUI to generate simulated keyboard inputs based on detected poses. The program captures live video through the webcam and processes every frame to identify body landmarks for translating particular movements into game actions: jumping, running, and shooting.

Key functionalities identified include pose detection for landmarks to appear only when high threshold values are recorded. It makes use of key points of shoulders, hips, and nose in such a way that the person is completely within the frame before the game logic comes into play.

Key Press Simulation: The PyAutoGUI package allows the program to simulate a key press and release, simulating interaction without real keyboard activity. The control system using a pose-based interaction introduces a new method of incorporating body movements into game interactions. Using the MediaPipe library helps ensure real-time processing and robust pose detection, so even dynamic user environments do not cause an issue. Multi-step logic involved in this - verify that the player is fully inside the frame and checks for multiple landmark positions - aids reliability of action detection.

VI. Challenges and Limitations

1. Lighting and Background Conditions : Poor lighting, as well as complex backgrounds, may deter pose detection accuracy. The limitation has the possibility of making misinterpretations of movements or missing detections.

2. Latency and Real-Time Performance : The program possesses adequate real-time performance, but might suffer delays on less powerful systems that come with the cost of computation involved in continuous video processing and pose analysis.

3. User Positioning: The system bases success on appropriate camera-distance as well as orientation and from users. The system might fail to identify actions duly if the user steps out of the frame or otherwise varies his pose significantly.

VII. Conclusion

The system demonstrates that pose-based control mechanisms are feasible and effective for gaming applications. This code works as an example of combining the computer vision and automation libraries to present highly immersive, interactive experiences that will be deepened with further development, robustness, and usability to include a wide range of applications other than gaming, like fitness tracking or interactive training simulation.

VIII. FUTURE SCOPE

1. Adaptive Thresholding : Implementing dynamic thresholds for action detection could improve the system’s adaptability to different users and environments.

2. Machine Learning Enhancements : Integrating a more complex machine learning model, such as a neural network, could improve the system’s ability to recognize a broader range of actions and adapt to user-specific movement patterns.

3. Feedback Mechanism : Adding real-time feedback on the screen (e.g., bounding boxes or guidance prompts) could assist users in maintaining proper positioning and improve interaction accuracy.

IX. References

[1] Trigueiros, Paulo, Fernando Ribeiro, and Luís Paulo Reis. "Hand gesture recognition system based in computer vision and machine learning." *Developments in medical image processing and computational vision* (2015): 355-377.

[2] Tan, Chiang Wei, Siew Wen Chin, and Wai Xiang Lim. "Game-based human computer interaction using gesture recognition for rehabilitation." In 2013 IEEE International Conference on Control System, Computing and Engineering, pp. 344-349. IEEE, 2013.

[3] Peng, Chao, Jeffrey Hansberger, Vaidyanath Areyur Shanthakumar, Sarah Meacham, Victoria Blakley, and Lizhou Cao. "A case study of user experience on hand-gesture video games." In 2018 IEEE Games, Entertainment, Media Conference (GEM), pp. 453-457. IEEE, 2018.

[4] Chan, Alvin TS, Hong Va Leong, and Shui Hong Kong. "Real-time tracking of hand gestures for interactive game design." In 2009 IEEE International Symposium on Industrial Electronics, pp. 98-103. IEEE, 2009.

[5] Ionescu, Dan, Bogdan Ionescu, Cristian Gadea, and Shahidul Islam. "A multimodal interaction method that combines gestures and physical game controllers." In 2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN), pp. 1-6. IEEE, 2011.

[6] Khan, Asharul Islam, and Salim Al-Habsi. "Machine learning in computer vision." *Procedia Computer Science* 167 (2020): 1444-1451.

[7] Zaina, Luciana, Elisa Castro, Suellen Martinelli, and Tiemi Sakata. "Educational games and the new forms of interactions: Exploring the use of hand gestures in a computational thinking game." *Smart Learning Environments* 6 (2019): 1-17.3.

[8] Hsiao, Hsien-Sheng, and Jyun-Chen Chen. "Using a gesture interactive game-based learning approach to improve preschool children's learning performance and motor skills." *Computers & Education* 95 (2016): 151-162.

[9] Liu, Jing, and Manolya Kavakli. "A survey of speech-hand gesture recognition for the development of multimodal interfaces in computer games." In *2010 IEEE International Conference on Multimedia and Expo*, pp. 1564-1569. IEEE, 2010.

[10] Saman, Oana, and Loredana Stanciu. "Hand Mobility Recovery Through Game Controlled by Gestures." In *2019 IEEE 13th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pp. 241-244. IEEE, 2019.

[11] Gosalia, Niyati, Priya Jain, Ishita Shah, Abhijit R. Joshi, Neha Katre, and Sameer Sahasrabudhe. "3D gesture-recognition based animation game." *Procedia Computer Science* 45 (2015): 712-717.

[12] Bikos, Marios, Yuta Itoh, Gudrun Klinker, and Konstantinos Moustakas. "An interactive augmented reality chess game using bare-hand pinch gestures." In *2015 International Conference on Cyberworlds (CW)*, pp. 355-358. IEEE, 2015.

[13] Gori, Ilaria, Sean Ryan Fanello, Giorgio Metta, and Francesca Odone. "All gestures you can: A memory game against a humanoid robot." In *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pp. 330-336. IEEE, 2012.