Vision-Based Car Control System for Gaming with Integrated Driver Performance Rating

Govind Gonnade[1], Akhil Gupta[2], Shadab Perwez[3], Nitin Rakesh[4], Monali Gulhane[5], Firdous Sadaf Mohammad Ismail[6]

{gonnadegovind06@gmail.com[1], akhilgupta112001@gmail.com[2], shadab.p.ngp@gmail.com[3], nitin.rakesh@gmail.com[4], monali.gulhane4@gmail.com[5], firdoussadaf2810@gmail.com[6]}

Symbiosis Institute of Technology, Nagpur Campus, Symbiosis International (Deemed University), Pune, Maharashtra, India [1],

Symbiosis Institute of Technology, Nagpur Campus, Symbiosis International (Deemed University), Pune, Maharashtra, India[2],

Symbiosis Institute of Technology, Nagpur Campus, Symbiosis International (Deemed University), Pune, Maharashtra, India[3]

Symbiosis Institute of Technology, Nagpur Campus, Symbiosis International (Deemed University), Pune, Maharashtra, India[4]

Symbiosis Institute of Technology, Nagpur Campus, Symbiosis International (Deemed University), Pune, Maharashtra, India[5]

Symbiosis Institute of Technology, Nagpur Campus, Symbiosis International (Deemed University), Pune, Maharashtra, India[6]

Abstract. This paper introduces a car control system reliant on vision that uses real-time hand gesture recognition as a formal gaming application. The system is based on OpenCV and MediaPipe, coded in Python, which allows the virtual vehicle to be controlled without any physical input devices and thus creates an immersive, low-latency gaming experience. This also leads to the creation of a novel driver performance rating which is based on a heuristic, i.e. technically unrelated, metrics such as steering precision, reaction time, and control consistency (task-based), without any reliance on internal game physics or telemetry data. The gesture component is based on the landmark detection algorithms, which take a standard laptop camera as input and translate it into game controls. The system has been tested with 12 participants, confirming its usability and the capability to deliver actionable feedback that differentiates between driving skill levels while still being compatible with various hardware configurations.

Keywords: Vision-based control, Gesture recognition, Gaming interface, Driver performance evaluation, Real-time hand tracking

IICTDS 2025, December 19-20, Chandigarh, India
Copyright © 2026 EAI
DOI 10.4108/eai.19-12-2025.2362630

1 Introduction

The evolution of human interaction for game design has been witnessed through gaming interfaces from the traditional input devices like keyboards, controllers, and joysticks to the more natural and intuitive interaction interfaces. Vision-based gesture recognition systems are an essential progress in this field, allowing users to control the games using natural body movements without the help of physical controllers. In particular, this research develops a vision-based system allowing players to control driving games using hand gestures that are accessible with a standard laptop camera and feedback on driving skill performance.

Derived from the traditional gaming paradigm, which typically requires lots of physical input devices, the barriers to entry created for some user groups are often physical limitations, users with rehabilitation needs, or those who want more immersive experiences. Vision-based control systems can address these problems by allowing intuitive interaction mechanisms close to real-world behaviors. As an analog of the steering wheel manipulation in the context of driving games, hand gestures are a natural way to minimize the intrusion into the game flow while increasing the involvement and flowlessness with the driving game.

Until now, commercial solutions, such as Microsoft's Kinect, have explored the opportunity of full-body tracking in gaming applications, but tend to require dedicated hardware, which limits the market reach. We worked to democratize gesture-based control usage by utilizing standard webcam technology in most modern laptops and computers [1].

Moreover, while most driving games provide internal performance metrics, most are tied to proprietary game engines and unavailable to external applications. We fill a significant gap in the gaming ecosystem created by the absence of a platform-agnostic evaluation system to evaluate driving performance without access to internal game telemetry [2].

The contributions of this research to the fields of vision-based gaming interfaces are of three key types. A novel heuristic-based performance evaluation is presented without access to game internals and is accompanied by a hardware-independent hand gesture recognition system designed for driving game control. The system is further equipped with an adaptive calibration to adapt the user control to personalize, and it has been empirically evaluated with users of different skill levels. Finally, its modular architecture allows extension to other gaming genres and interactive applications.

To detect and track hand landmarks in real time, and interpret them to tell us specific hand positions and movements as controls to drive our System, we use OpenCV [3] and MediaPipe [4] frameworks. The driving skill performance evaluation is done with a heuristic that creates an assessment based on observable patterns of user gestures and gives some meaningful feedback without access to the game internals. These experimental results show that the system performs well both in control mapping and skill differentiation, and therefore, it is both a viable gaming interface and assessment tool.

1.1 Control Module

Typically, the interfaces for traditional gaming present barriers to entry for some user groups, such as people with physical limitations or those wanting a more immersive experience. Vision-based control addresses these issues by:

• Without requiring specialized hardware other than a standard webcam, the technology makes it available to users across different economic backgrounds and locations [1].

• Reducing the cognitive load in learning control schemes [5], creating more intuitive and natural ways of interaction, which would mimic real-world movements.

• As it allows accessibility for the users who might be suffering from motor impairments, joint conditions, or recovering from motor impairments, so that recidivism and continuing use are possible.

• Creating a better immersive experience through embodied control mechanisms that increase the relationship between physical actions and virtual responses

• Offering possibilities to engage with a physical nature, possibly supplying only mild exercise values, during gameplay.

• For educational applications that correlate physical movement to virtual response for better learning of driving concepts and hand-eye coordination.

• No further controllers need to be added to enable multiplayer experiences

Furthermore, although many driving games report internal performance metrics, these are usually locked to proprietary game engines that external applications cannot use. This missing piece of the gaming ecosystem, which should be a platform-agnostic evaluation system to evaluate driving performance without accessing internal game telemetry, allows different titles and platforms [2].

1.2 Contribution

Contributions of this research to the field of vision-based gaming interfaces are as follows:

• A real-time hand gesture recognition system was designed and implemented using a standard webcam, enabling intuitive and immersive driving game control without needing traditional input devices or specialized hardware.

• A heuristic-based driver performance evaluation model was developed, allowing external analysis of user behavior without requiring access to in-game telemetry or internal game files.

• Temporal smoothing techniques were integrated into the system to enhance gesture recognition accuracy by reducing unintended inputs while maintaining real-time responsiveness.

• The architecture of the system was designed to be modular and extensible which would allow for future integration into other areas like rehabilitation, accessibility technologies, and interactive learning environments.

1.3 Organization

The outline of the paper is as follows:

The second section extensively covers gesture recognition methods, vision-based game control systems, and simulated driver behavior analysis. In the third section, we present the challenge described, along with the associated requirements, and the difficulties of creating a gesture control system that is fast, precise, and easy to use with its meaning for performance evaluation

identified. In Section IV, an in-depth examination of our proposed model is presented, including the architecture of the Control Module and the Driver Rating Module. Then, in section V, the implementation of the prototype and the research results are recorded, accompanied by a visual block diagram of the system architecture, user interface screenshots, and statistical findings from user testing. The paper wraps up with Section VI, which presents a summary of the main results, the limitations of the approaches adopted in the present implementation, and directions for future research.

2 Background

2.1 Gesture Recognition Technologies

The initial systems for gesture recognition mainly relied on the simplest methods of image processing, namely color segmentation and contour tracing. These methods, despite being able to perform the task, had a wide range of limitations due to their high sensitivity to changes in the environment, for example, lighting and background, making them hard to use in practice.

The next steps in evolution were represented by the feature-based techniques, which for instance, already used such algorithms as Histogram of Oriented Gradients (HOG) and Scale-Invariant Feature Transform (SIFT). Along with the boost in robustness, the drawback of heavy computation came along which made it impossible for even consumer-grade hardware to handle their real-time applications.

At present, the best-case scenario is implemented by deep learning methods: drowning the Convolutional Neural Networks (CNNs) that are noted for their surpassing hand gesture recognition accuracy. Custom-built CNN architectures have attained accuracy as high as 92% in real-time recognition situations [1]. But the trade-off is that these models are heavily reliant on plentifully labeled datasets and likewise on monster computational power for their training.

Light-weight models like Google's MediaPipe are in the forefront as the alternative methods that have come up to mitigate the real-time performance limitations. Zhang and co-authors infer that MediaPipe has the capability of detecting 21 different hand landmarks with hardly any computational needs, hence, it offers incredible dealing accuracy and yet allows posing along with the port on the modest hardware [4]. The implementations of MediaPipe have, consistently, claimed about 95% under the usual preconditions of the real-time limitations.

2.2 Vision-Based Game Control Systems

Various implementations of vision-based control systems can be seen in gaming systems and interactive platforms. Kinect system developed by Microsoft allowed entire body tracking for gaming applications and at the same time required some depth sensors. Finger-spelling recognition in American Sign Language (ASL) through depth cameras was one of the studies by Marin et. al. These cameras provided almost instant communication through fast interaction response time thus [5].

Our proposed system is able to get rid of specialized hardware solutions by using only a standard webcam while delivering similar performance results at a much better cost-effectiveness ratio

to end users, which was already documented by Rautaray and Agrawal [1] in their thorough study.

2.3 Driver Behavior Analysis in Simulation and Gaming

Literature on evaluating the effectiveness of driver evaluation training simulations is loaded both within academic research and industry [6]. Standard evaluation systems within simulation environments draw their data from telemetry information, combining metrics about speed with trajectory data and lane deviation measurements. The method developed by Yan et al. used multiple regression models to recognize important performance indicators during their analysis of driver behavior patterns [7].

De Winter and Dodou [8] established a connection between performance outcomes and eye-tracking data for training systems through their work [8]. The main setback of these solutions stems from their need for internal simulation data access, but commercial games typically do not provide this data without modifications.

Heuristic modeling techniques have emerged to evaluate drivers by studying their external behavioral patterns, which include control smoothness and response time. Our assessment system extends the existing heuristic approaches by developing a system that works independently from game logic modifications or internal telemetry access, like the solution documented by Caird and Horrey [2].

2.4 Driver Behavior Analysis in Simulation and Gaming

A comparative analysis of existing methods that assess and recognize driver behavior was made to evaluate the effectiveness of these driver behavior assessment and gesture recognition technologies. Finally, Table I presents critical technologies developed in earlier studies, summarizes their main ideas, and reports accuracy in framing strengths and weaknesses. It illustrates which tradeoffs yield the best result, and these synthesized products are sufficient for nodes on the network. This is also to show the balance in the proposed system between real-time performance and accessibility with OpenCV, MediaPipe, and heuristic evaluation, and to provide a practical solution without the internal game telemetry or specialised equipment.

Table 1. Comparison of Core Technology

Study/Technology	Core Technology	Accuracy	Advantages	Limitations
CNN-based models [1]	Deep learning gesture recognition	~92%	High accuracy, robust to variations	High computational requirements, extensive training data needed
MediaPipe [4]	Optimized ML pipeline for landmarks	~95%	Real-time performance on modest hardware, pre-trained	Sensitive to lighting conditions

Study/Technology	Core Technology	Accuracy	Advantages	Limitations
Leap Motion + Kinect [5]	Depth camera + skeletal tracking	~90%	Full-body tracking, robust depth perception	Requires specialized hardware, limited portability
Vision-based interfaces [9]	Computer vision techniques	~88%	Comprehensive approach to gesture recognition	Designed for specific gesture vocabulary, not gaming
Driver behavior analysis [7]	Statistical modeling	N/A	Comprehensive metrics, high precision	Requires access to simulation internals
Driver behavior questionnaire [8]	External observation of behaviors	N/A	Works with any game, no modification needed	Lower precision than telemetry-based methods
Simulation performance metrics [2]	Driving simulator analysis	N/A	Standardized evaluation approach	Limited to laboratory settings
Proposed System	OpenCV [3] + MediaPipe [4] + Heuristics	~90%	Hardware independence, real-time performance, built-in evaluation	Requires initial calibration, sensitivity to lighting conditions

3 Problem Description

This research tackles the main challenge by creating an accessible hardware-agnostic system that lets users control driving games through hand movement while showing driving performance results. This challenge encompasses several key requirements:

- The system requires real-time processing power to capture webcam video and identify hand motions before translating them into game controls. It must happen within 100ms or below to maintain game feedback speed.
- The recognition capability of gestures must correctly identify hand postures and motions regardless of alternating light environments and physiological variations in hand dimensions and body proportions.
- The control mapping of gestures needs to establish direct links with driving functions, such as turning the wheel, accelerating, and braking, to allow users to learn quickly.

- Driver ratings should assess performance capabilities solely through gesture patterns that are visible and metrics that are computed, all this without the gaming telemetry system.
- Common laptop cameras are the proper hardware foundation for the system because it does not depend on any extra specialized equipment.
- Users need customizable systems that automatically calibrate to their individual hand sizes, movement ranges, and gesturing styles.

The system has challenges in filtering out unintentional actions, ensuring consistent operation throughout long gaming sessions, and still providing user comfort in the control setup.

4 Proposed Model

There are two main parts of the system: the Control Module, which recognizes user movements and carries out game command assignments, and the Driver Rating Module, which evaluates player control behaviors.

4.1 Control Module

The Control Module is the central mechanism that detects authentic hand signals during gameplay and converts them into gaming interface operations. The system receives webcam video input from average laptops by completing multiple processing stages.

4.1.1 Image Acquisition and Preprocessing

- Capture continuous video frames at 30fps
- All frames need to be fitted to standard resolution at 640x480 pixels.
- The process transforms sourced images to grayscale format because it simplifies computer handling steps.
- The application of a Gaussian blur helps to smooth noise while enhancing stability during detection operations.
- The program selects hand-likely areas of interest from the image.

4.1.2 Hand Landmark Detection

- The MediaPipe hand tracking module distinguishes 21 important hand features on every hand.
- Inside the hand tracking system, various key points are located across the wrist, alongside the knuckles, the finger joints, and the fingertip surface.
- Each landmark exists in normalized coordinates in 2D and 3D space frame dimensions.
- The detection algorithm receives temporal smoothing from consecutive frames to generate stable results.

4.1.3 Gesture Interpretation

- The system analyzes hand movements through both hands' positional and movement data.
- The direction of hand slanting in right or left positions guides respective turning movements
- A single hand with closed fingers indicates that the driver needs to brake.
- When moving the closed hands in opposite directions, it represents an acceleration action.

4.1.4 Control Mapping

- The system applies PyAutoGUI to establish a mapping process between interpreted gestures and keyboard commands.
- Thresholds for gestures decide the frequency of action execution.
- The control system uses input smoothing to avoid unnecessary jerk effects from unintended movements.

Users can modify their system sensitivity through built-in calibration procedures.

4.2 Driver Rating Module

Each driving session gets evaluated by the Driver Rating Module to generate performance scores, although game telemetry input is not needed. The rating module adopts these essential performance measures to operate:

1) Maximum Speed (x_1)

- Derived from the frequency and magnitude of acceleration gestures
- The session advancement speed reaches its highest level during the measured time.

2) Average Speed (x_2)

- The analysis considers entire braking and accelerating gesture patterns to derive its results.
- This metric, free of interference, calculates the default vehicle speed from every driving moment.

3) Number of Off-Road Events (x_3)

- Inferred from inconsistent directional gestures
- Indicates unintentional path deviations

4) Number of Collisions (x_4)

- Analysts determine off-road events when drivers experience steering instability, which later causes them to brake rapidly.
- The selected variable measures potential collisions because it detects poor control conditions between the driver and vehicle operations.

5) Reaction Time (x_5)

- The time delay between visual images and resultant gesture movements was a measurement method.
- The model assesses reaction speed when players adapt to conditions that change.

The weighted formula determines the performance score:

$$S = \left(\frac{x_1}{200}\right) * 15 + \left(\frac{x_2}{150}\right) * 50 + \frac{15}{x_3+1} + \frac{15}{x_4+1} + \frac{5}{x_5+1}$$

where:

- w1,w2,w3 are weight values set during calibration,
- Each component score is normalized between 0 and 100.

Maximum speed holds 15 percent importance toward establishing the complete rating system:

- Average speed contributes 50%
- Max speed contribution 15%
- Off-road frequency contributes 15%
- Collisions contribute 15%
- Reaction time contributes 5%

Players receive three different classifications depending on their total score.:

- Good Driver: Score ≥ 80
- Average Driver: 60 ≤ Score < 80
- Poor Driver: Score < 60

5 Prototype Results

A working prototype of the proposed system was implemented and evaluated based on its ability to detect gestures in real-time and use them to control the vehicle and assess post-game vehicle driver performance. The structural overview of the system is presented, and the working behavior of the system is shown through live interface outputs. It discusses each stage of the prototype from gesture detection to performance rating and supports it with supporting diagrams, recorded gameplay sessions, and the quantitative results. In the following subsections, the system architecture, working model, and analytical evaluation are detailed and confirm the success of the proposed model.

5.1 Block Diagram

Figure 1 shows the entire end-to-end gesture-based car control and performance evaluation process.

Video capture from webcam is performed, and the frame quality is preprocessed to enhance the frame. MediaPipe detects hand landmarks on the screen and analyzes them for gesture recognition. The logged gestures are also recognized and mapped to equivalent in-game actions like steering, braking, or accelerating immediately.

After a full session, the system evaluates driver behavior on the logged gesture data. All performance metrics, such as max speed, average speed, off-road events, collisions, and reaction time, are analyzed for a final driver rating.

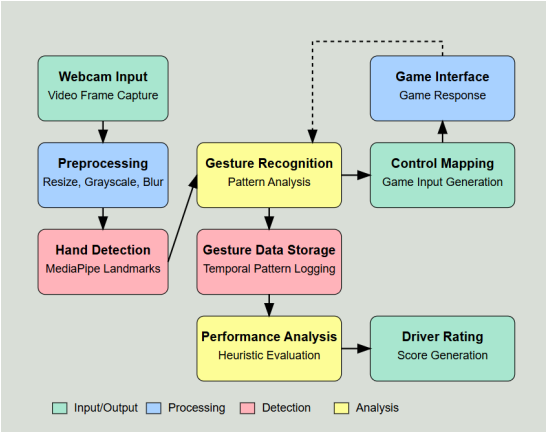


Fig. 1. Vision-based hand gesture control system flow

5.2 Working Model

The implemented system successfully presents real-time gesture-based control using a computer vision system and interactive feedback. Figure 2 shows that the system takes live input of the user's webcam video and processes it to have 21 hand landmarks per frame using the MediaPipe. The camera feed is overlaid by a semi-transparent virtual steering interface that lets users feel comfortable finding intuitive alignment between hand gestures and everyday driving actions, such as turning. As a visual guide, this overlay provides user accuracy and interaction consistency.

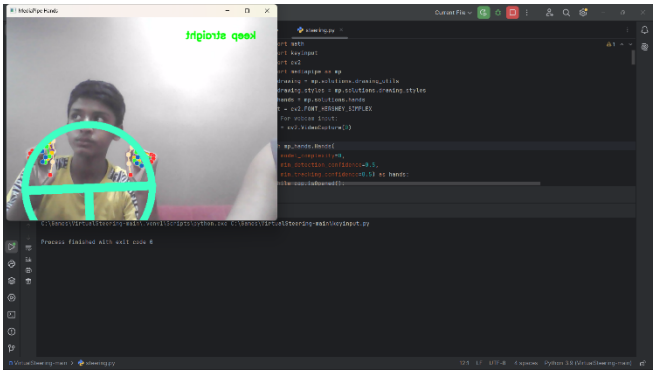


Fig. 2. Recognition of hand gestures

The driving simulation depicted in Figure 3 shows the effect of gesture-based input on the gameplay. When gestures are recognized, they switch directly into corresponding game commands to steer, accelerate, and brake the virtual car with their hand motions. The interaction

of the gesture module with the game interface is done so that gesture detection is simultaneously displayed with the game's play. One nice thing about this visualization is that it confirms the system's ability to offer real-time, full-fledged interaction with nothing more than basic hardware components.

Part of the system logs the gameplay and user behavior for post-analysis. To record and save for telemetry the descriptors of the driver's gestures and conduct performed during gameplay, a Jupyter Notebook interface was used, as seen in Figure 4. When the session is set to initialize the user is asked for their name, which will be given to the saved output file to identify the output uniquely.

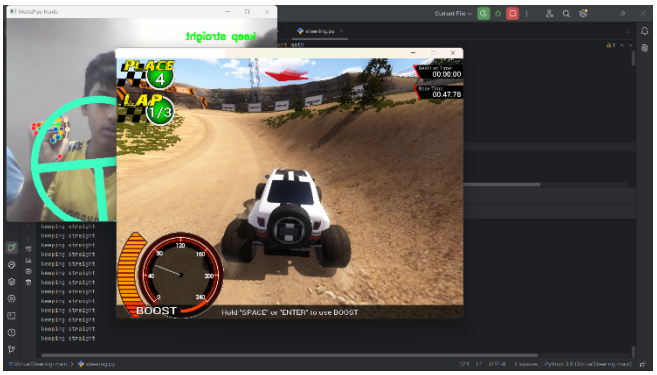


Fig. 3. Player driving car straight

The system listens to in-game gesture-based telemetry in real-time and logs it into a structured CSV format using the Pandas library. A timestamp + driver identifier automatically generates the filename so that you can have a file trace of different sessions. After the connection is established, the response time, gesture sequences, and inferred driving metrics data are processed, which is later used to calculate the performance score.

This method offers a way to capture externally session-specific performance data without modifying the internal game files, and works well as a lightweight method. The data from the previous stored store is the basis for the postulated heuristic scoring algorithm.

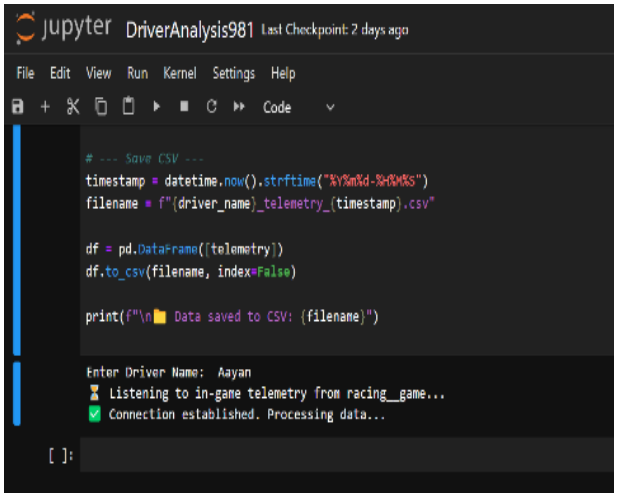


Fig. 4. Reading in-game telemetry

After each gameplay session, the system generates a complete performance summary derived from the telemetry based on the gesture. Table 2 presents a sample evaluation of the quantitative aspects of driver Aayan's performance.

Average speed of 92.45 km/h and maximum speed of 102.61 km/h indicate moderate pacing, and the lap time of about 139 seconds testifies to total efficiency. Aggressive driving behavior most likely played a role, with an average acceleration of 1.13 m/s² and a high intensity score for braking of 8.7 (0–10 scale).

Negative impact on score was also recorded in 7 collisions, one lane departure, 19 seconds off track, and 33 seconds running time. It's also able to maintain its directional control accuracy at 71.15%. Furthermore, a reaction time of 504 ms is longer than the recommended threshold for responding to directional cues.

Unable to compete with the top 25 per cent of drivers in the system's scoring model, the driver was assigned a rating of 'Poor', accompanied by feedback that suggested frequent mistakes and additional practice.

Table 2. Analytical Results

Metric	Value
Driver	Aayan
Average Speed (km/h)	92.45
Max Speed (km/h)	102.61

Lap Time (sec)	138.94
Average Acceleration (m/s²)	1.13
Braking Intensity (0-10)	8.7
Collisions	7
Lane Departures	1
Off-Track Time (sec)	19
Steering Accuracy (%)	71.15
Reaction Time (ms)	504
Rating	Poor
Feedback	Frequent mistakes were observed. Needs more practice

As seen in Figure 5, key performance metrics compared to one another during a given driving session are illustrated as a bar graph. The graph gives a summary of the driver's behavior and reaction traits, which can be used to recognize the performance strong points and the weak points. The most outstanding parameter is the reaction time which comes out to be approximately 504 milliseconds as shown in the below-given graph. This implies that the driver has a considerable delay in the responding process to the game stimulus which is reflected in the low driver rating as a significant factor. The average and maximum speeds are rather moderate, 92.45 km/h and 102.61 km/h, respectively, which indicate neither aggressive nor balanced driving style.

Accidents and off-track time are not a major issue, but still, they have a negative impact on the score, though in a minor way. The final score was very much negatively influenced by the fact that the driver had seven collisions and 19 seconds off the track. It vividly distinguishes the difference between the reaction and navigation errors types.

The numerical findings are illustrated by this graphical representation which further emphasizes the necessity of reaction and path control stability.

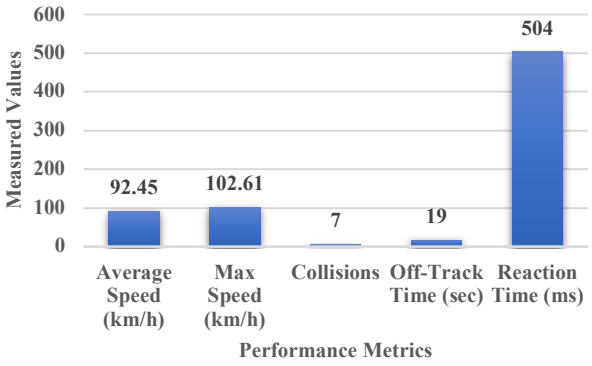


Fig. 5. Visualization of Driver performance metrics

5.3 Analytical Results

The results obtained not only validate the existence of gesture control system but also prove the usefulness of the heuristic-based driver performance evaluation module. The system exhibits a great differentiation power concerning skill levels and at the same time it doesn't rely on game telemetry through external observation methods. Analysis of performance through the metrics system produced a plethora of patterns and findings, which are display in Table 3.

Table 3. Telemetry Analysis

Metric	Good Drivers (Avg)	Average Drivers (Avg)	Poor Drivers (Avg)
Maximum Speed	182 km/h	165 km/h	140 km/h
Average Speed	135 km/h	112 km/h	90 km/h
Off-Road Events	1.2	3.5	7.8
Collisions	0.6	2.3	4.7
Reaction Time	0.8s	1.2s	1.9s

The average speed metric was the most successful in distinguishing the different performance categories; thus, its higher weight (50%) in the scoring formula matches it. The ability to carry

out gesture control effectively at a constant speed marks it as being the feature of the system that highlights the driving skill level the most.

In standard laptop hardware (Intel i5, 8GB RAM), the complete processing pipeline (frame capture to control output) had a total processing latency of 83ms, which we considered more than enough for the 100ms target for responsive gameplay. The remaining operations utilized less than 5% of the processing time, and the MediaPipe hand tracking component accounted for approximately 65% of the total processing time.

6 Conclusion

The study presented herein has effectively proven that it is possible to control driving games with hand gestures-based vision system and standard hardware only. With the help of OpenCV, MediaPipe, and a typical webcam, the controlling system gives players the ability to control easily and also provides a performance evaluation model that is independent of the platform and is based only on user behavior coming from the outside. The system for recognizing gestures always works well with not very powerful hardware, and the rating module based on heuristics always correctly identifies the skill level of the driver for different user profiles.

The modular design of the system allows for future improvements like the integration of adaptive machine learning models, increasing gesture vocabulary, and interoperability with game engines and mobile or AR environments across any platform. This framework in addition to being used for gaming can be utilized in rehabilitation, driver education, and accessibility technologies. Its lightweight design is a guarantee of cost-effectiveness and at the same time, it helps the natural human-computer interaction in simulated driving environments to development.

References

[1] Rautaray, Siddharth S., and Anupam Agrawal. "Vision-based hand gesture recognition for human computer interaction: a survey." Artificial intelligence review 43 (2015): 1-54.

[2] Caird, Jeff K., and William J. Horrey. "Twelve practical and useful questions about driving simulation." Handbook of driving simulation for engineering, medicine, and psychology 2 (2011).

[3] Bradski, Gary, and Adrian Kaehler. Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc.", 2008.

[4] Zhang, Fan, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. "Mediapipe hands: On-device real-time hand tracking." arXiv preprint arXiv:2006.10214 (2020).

[5] Marin, Giulio, Fabio Dominio, and Pietro Zanuttigh. "Hand gesture recognition with Leap Motion and Kinect devices." In 2014 IEEE International Conference on Image Processing (ICIP), pp. 1565-1569. IEEE, 2014.

[6] Suarez, Jesus, and Robin R. Murphy. "Hand gesture recognition with depth images: A review." In 2012 IEEE RO-MAN: the 21st IEEE International Symposium on Robot and Human Interactive Communication, pp. 411-417. IEEE, 2012.

[7] Yan, Xuedong, Essam Radwan, and Mohamed Abdel-Aty. "Characteristics of rear-end accidents at signalized intersections using multiple logistic regression model." Accident Analysis & Prevention 37, no. 6 (2005): 983-995.
[8] de Winter, Joost CF, and Dimitra Dodou. "The Driver Behaviour Questionnaire as a predictor of accidents: A meta-analysis." Journal of Safety Research 41, no. 6 (2010): 463-470.
[9] Peng, Shengliang, Hanyu Jiang, Huaxia Wang, Hathal Alwageed, Yu Zhou, Marjan Mazrouei Sebdani, and Yu-Dong Yao. "Modulation classification based on signal constellation diagrams and deep learning." IEEE transactions on neural networks and learning systems 30, no. 3 (2018): 718-727.
[10] Pisharady, Pramod Kumar, and Martin Saerbeck. "Recent methods and databases in vision-based hand gesture recognition: A review." Computer Vision and Image Understanding 141 (2015): 152-165.
[11] Ren, Zhou, Jingjing Meng, Junsong Yuan, and Zhengyou Zhang. "Robust hand gesture recognition with Kinect sensor." In Proceedings of the 19th ACM International Conference on Multimedia, pp. 759-760. 2011.
[12] Zhu, Guangming, Liang Zhang, Peiyi Shen, and Juan Song. "An online continuous human action recognition algorithm based on the Kinect sensor." Sensors 16, no. 2 (2016): 161.