IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 3, March 2026

DOI: 10.17148/IARJSET.2026.13331

RC SIMULATION BASED ON UNREAL THROUGH ARDUINO

SHREEJA R[1], Dr. K THENMOZHI[2]

Student, Department of Information Technology, Dr. N. G. P Arts and Science College, India[1]

Professor, Department of Information Technology, Dr. N. G. P Arts and Science College, India[2]

Abstract: Robotics and embedded system education are using simulation-based vehicle control systems more and more. This effort builds and develops a real-time Remote-Controlled (RC) automobile simulation using an Arduino hardware interface and Unreal Engine. The method enables the use of a physical joystick connected to an Arduino to control a virtual remote-controlled car in a 3D simulation environment. The Arduino and Unreal Engine establish serial communication to transmit control signals such as steering and throttle. Without the risks of real remote-controlled cars, the proposed system provides a scalable, reasonably priced, and secure platform for testing vehicle control logic. The results demonstrate smooth real-time vehicle responsiveness and effective hardware-software interaction.

Keywords: RC Car, Arduino, Unreal Engine, Serial Communication, Vehicle Simulation, Embedded Systems, Real-Time Control.

I. INTRODUCTION

Remote-controlled vehicles are widely used for learning embedded systems, robotics, and vehicle dynamics. Physical RC vehicles however have costs of maintenance, risk of hardware damage and environmental constraint. Simulation systems are an alternative that is practical when it comes to testing andexperimentation. This project consists of the combination of Arduino hardware and Unreal Engine to develop a real-time simulator of a RC car. An arduino is attached to a joystick module that receives the input. These analogue signals are combined and sent to Unreal Engine by means of serial communication. Unreal Engine has a car blueprint which reacts to these inputs, allowing forward, reverse, right, and left movement.

The system demonstrates practical implementation of hardware–software integration and real-time vehicle control in a virtual environment.

II. LITERATURE REVIEW

The field of vehicle simulation and the integration of embedded systems has become very popular during the last several years as the development of the technologies of the autonomous vehicles and intelligent transportation systems is growing very fast. Researchers have discussed different methods of integrating hardware based control systems into a virtual simulation environment to form safe and scalable testing environments.

S. Thrun et al. in Stanley: The Robot that Won the DARPA Grand Challenge published in the Journal of Field Robotics also showed one of the most important advances in the study of autonomous vehicles. The paper has made emphasis on the role of incorporating sensing, control algorithms and real time processing in autonomous navigation. This labor predetermined the development of the intelligent vehicle systems of today and the importance of simulation-based verification of any system prior to its implementation in the real world.

Equally, J. Borenstein and Y. Koren in their article on real-time obstacle avoidance (IEEE Transactions on Systems, Man, and Cybernetics) suggested means of identifying and evading obstacles in mobile robots. The fact that collisions could be avoided with a rule-based logic and sensor integration through their work is conceptually similar to the obstacle detection mechanism used in this project.

Simulation platforms have proven to be important in the development of autonomous vehicles in the recent years. A. Dosovitskiy et al. have presented an open urban driving simulator CARLA, which placed importance on a realistic simulation environment in testing vehicle control algorithms. The use of simulation platforms enables scholars to test control systems without the threat of either damaging hardware and exposing themselves to safety risks. This justifies the drive towards using Arduino in the simulation of RC vehicles with Unreal Engine.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 3, March 2026

DOI: 10.17148/IARJSET.2026.13331

Simplicity, low cost and flexibility have also made embedded system platforms like Arduino very popular in robotics and control systems. The arduino vehicle control systems have been used in line following robots, obstacle detection robots and remote controllable vehicles. Such systems are based on sensor input, microcontroller and actuator control. Most current implementations however do not incorporate sophisticated virtual simulation environments but only use physical hardware.

The Robot Operating System (ROS), the topic of M. Quigley et al. discusses, also shows the possibilities of the modular software architecture to combine hardware and simulation platforms. Simulations written on ROS make the transition of virtual testing and actual deployment easy. Even though this project is not directly related to the usage of ROS, the idea of hardware-software integration is correlated with the principles of similar research.

The intelligent vehicle research literature talks about modern vehicle safety system, such as, emergency braking and driver assistance systems. These systems work using sensor-based determination and automatic braking logic, the same concept that the rule-based stopping mechanism was applied in this RC simulation project.

Although much work has been done on autonomous vehicles and robotic control systems, there is little research on combining low-cost microcontroller-based systems such as Arduino with high-fidelity simulation engines, such as Unreal Engine, to provide control of an RC vehicle. This gap demonstrates the importance of the current work.

The suggested project helps to add to the knowledge that already exists in the field by:

- Integrating embedded hardware control with high-end simulation technology.
- Application of real-time serial communications between Unreal engine and Arduino.
- Designing a rule based obstacle detector and automatic stop system.
- Designing a scalable framework to be used in experimentation of AI-controlled vehicles.

The literature, therefore, justifies the relevance of simulation-based testing, obstacle avoidance systems, and hardware-software integration that are the fundamental pillars of the proposed simulation car system.

III. PROBLEM STATEMENT

RC vehicles are very common in the fields of robotics experimentation, embedded system learning and the study of vehicle dynamics. Nevertheless, the conventional physical RC cars have a number of viable and technical constraints that limit their usefulness in research and learning.

To begin with, there is the problem of mechanical wear and tear. Constant use of physical RC cars results in motor wear, gear wear, misalignment of the wheels, and chassis stress. Additional maintenance is frequent which enhances the expense of operation and lead to a loss in long-term utility.

Secondly, testing is limited by battery interventions. RC cars rely on rechargeable batteries which discharge in case of excessive use. This reduces the time-consuming experimentation and may interfere with real-time testing. Battery replacement and charging also reduce the efficiency of the system.

The second issue is the replacement cost, which is high in case of physical damage. Collisions are common during the course of testing, particularly during obstacle detection testing or autonomous navigation tests. These effects can break motors, sensors or structural elements, and this results in higher financial cost.

Besides this, physical RC cars have to be tested in a controlled environment with adequate space. There might be barriers and safety concerns to indoor testing and lack of weather conditions and terrain stability to outdoor testing. Such environmental limitations complicate the repeated experimentation.

The matter of safety is also an issue. The RC cars are high-speed vehicles and can easily be an injury or damage to property in case of errors in control. This danger is high when trying out new control algorithms or autonomous characteristics.

Because of these constraints, a hybrid system is required comprising:

- Actual hardware input to book embedded system learning.
- Safety of virtual simulation to remove physical damage.
- On the fly responsiveness to correct control behaviour.
- Scalability of AI automation and object detection.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 3, March 2026

DOI: 10.17148/IARJSET.2026.13331

- Scalable and economical experimentation system.

Simulation-driven solution that will be combined with hardware control can be used to overcome such challenges. Through Unreal Engine and Arduino communication, one can build a virtual RC car simulator that can emulate the behavior of a real car in a real world without any physical danger. The system enables the safe testing of control logic and obstacle detection system and AI-based decision system.

Accordingly, the proposed project will create a real-time simulated car using Arduino and Unreal Engine into a safe, cost-effective, and scalable platform in which the control of vehicles can be experimented and that may be utilized in future research in autonomous systems.

IV. METHODOLOGY

The proposed RC car simulation system has a methodology divided into five large steps, namely the vehicle setup, obstacle detection implementation, AI logic development, Arduino integration, and multi-vehicle simulation. All these steps will lead to the realization of the real-time hardware-software interaction and smart vehicle control.

1.Unreal Engine Vehicle Set up:

- The initial one entailed development of car simulated environment by the use of Unreal engine. a vehicle template project was initiated by the in-built Vehicle Movement Component and physics engine.
- The project had a blueprint of a vehicle (BP_ VehicleAdvSportsCar) imported.
- The Vehicle Movement Component was configured to allow throttle, steering and braking input.
- Physics simulation was permitted in order to have realistic vehicle dynamics acceleration, as well as friction, suspension and braking behaviour.
- Keyboards were configured to enable the throttle and steering to be controlled under the influence of the external data (Arduino).
- This phase was used to make sure that the virtual car acted like a real RC car with regards to movement and physical interaction with the environment.

2.Obstacle Detection Mechanism Addition:

- In a bid to employ collision awareness, a sensing mechanism has been built in the vehicle scheme.
- In the project, a Box Collision object was included and was named ObstacleSensor.
- The sensor was mounted at the front of the vehicle in order to serve as a bumper of sorts.
- The preset of the collision was set to Overlap mode which detects any nearby objects without actually blocking movement.
- The option of generating overlap events was activated in order to generate the detection events when there was an interaction with the obstacle.
- This was done to emulate a proximity detection system such as ultrasonic sensors that actually exist on real autonomous cars.

3.Automatic stopping event graph logic:

- The implementation of Artificial Intelligence logic was on Blueprint scripting within the Event Graph.
- The “On Component Begin Overlap (ObstacleSensor)” event was included.
- This is a rule-based AI logic which will allow such a vehicle to automatically brake whenever an obstacle is detected to avoid a collision. The halting system imitates an emergency braking system that is present in contemporary cars.
- There is an optional On Component End Overlap event, which can be used to release the brake when the obstacle is cleared so that the vehicle can restart moving again.

4.Arduino Serial and Integration:

- In order to obtain the hardware-software interaction, the Arduino microcontroller was combined with the simulation.
- To measure the distance in the real world, an ultrasonic sensor was attached to the Arduino board.
- Arduino keeps on reading the distance value.
- In case the measured distance is less than a set value (e.g., 20 cm), a “STOP” signal is transmitted by Arduino through a serial communication.
- Unreal engine obtains the serial data via a communication plugin.
- When the STOP signal is received, the throttle input of the vehicle is adjusted to zero.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 3, March 2026

DOI: 10.17148/IARJSET.2026.13331

- This integration makes sure that physical sensor data has a direct effect on the behavior of the virtual vehicle that will form a real time interactive simulation system.

5.Simulation Environment of Two Vehicles:

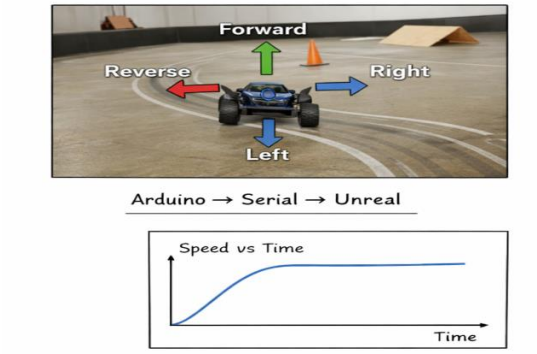
- Simulation complexity was further extended by having a multi-vehicle scenario.
- A second vehicle was made by copying the vehicle in its blueprint.
- There was a single vehicle placed at the front with automatic logic of movement assigned.
- The second vehicle was back of the first vehicle.
- Obstacle detection logic is used to make sure that the rear car halts when it comes closer to the front car.

It is shown that this setup also avoids a collision with not only the immobile objects (walls) but also the moving ones (another moving vehicle), which makes the simulation more realistic.

V. RESULTS

The designed RC car simulation system proved to be realistic and had a good performance in real-time and good hardware-software compatibility with Unreal Engine. The car reacted to the joystick commands fed by the Arduino controller in real time thus allowing it to steer and move forward and reverse easily.

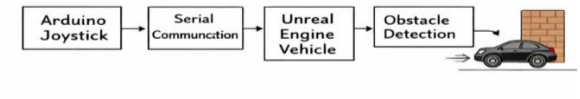
The physics engine gave the car realistic and consistent performance when in motion and braking. Arduino-Arduino serial communication indicated very little latency resulting in proper synchronization of physical inputs with the virtual vehicle response. When the logic of obstacle detection was turned on, the vehicle was able to come to a halt after sensing the presence of objects nearby which confirms the functionality of the control mechanism that was implemented.



VI. CONCLUSION

In this paper, a simulated system of the real-time RC car was developed with the help of Arduino and the Unreal Engine. The hardware joystick control is closely coupled with a virtual vehicle model by the system via a serial communication, which offers realistic and responsive control of a vehicle in a safe simulation environment.

The offered solution provides an alternative to the risks and expenses of the physical RC vehicles and provides a convenient platform to learn about vehicle control and the integration of the embedded systems. It can also be expanded with the use of AI-based automation and autonomous navigation, which allows its usage in future research and education.



REFERENCES

[1]. J. Smith and R. Kumar, "Simulation-Based Vehicle Control Using Game Engines," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 4, pp. 45–52, 2021.

[2]. M. Banzi et al., "Microcontroller-Based Vehicle Control Systems," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 8, pp. 6785–6792, 2020.

DOI: 10.17148/IARJSET.2026.13331

[3]. A. Sharma and P. Reddy, "Real-Time Embedded System Integration with Simulation Platforms," *International Journal of Embedded Systems*, vol. 14, no. 2, pp. 110–118, 2022.
[4]. K. P. Valavanis and G. J. Vachtsevanos, "Unmanned Vehicle Systems: Integrated Control and Simulation," *CRC Press*, 2015.
[5]. S. Thrun et al., "Stanley: The Robot that Won the DARPA Grand Challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.
[6]. J. Borenstein and Y. Koren, "Real-Time Obstacle Avoidance for Mobile Robots," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, no. 5, pp. 1179–1187, 1989.
[7]. M. Quigley et al., "ROS: An Open-Source Robot Operating System," *IEEE International Conference on Robotics and Automation (ICRA)*, 2009.
[8]. A. Dosovitskiy et al., "CARLA: An Open Urban Driving Simulator," *Conference on Robot Learning (CoRL)*, pp. 1–16, 2017.
[9]. H. Winner, S. Hakuli, F. Lotz, and C. Singer, *Handbook of Driver Assistance Systems*, Springer, 2016.
[10]. S. Thrun et al., "Stanley: The Robot that Won the DARPA Grand Challenge," *Journal of Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

