International Journal of Creative and Open Research in Engineering and Management
ISSN: 3108-1754 (Online)
Volume 02 Issue 11 November-2026 | Impact Factor: 3.5



Traffic Controller: An Interactive 3D Traffic Management Game

Pradeep K. V., Nishanth R., Bhagyashree H. G., Pavan J. T.
Department of Computer Science and Engineering Akshaya Institute of Technology, Tumakuru, Karnataka, India, Academic Year 2026-2027

How to Cite this Article:
V., P. K., R., N., G., B. H. & T., P. J. (2026). Traffic Controller: An Interactive 3D Traffic Management Game. International Journal of Creative and Open Research in Engineering and Management, 02(11), 1-9. https://doi.org/10.55041/ijcope.v2i11.148

License:
This article is published under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author(s) and the source are credited.
© The Author(s). Published by International Journal of Creative and Open Research in Engineering and Management.





https://doi.org/10.55041/ijcope.v2i11.148

***Abstract*—Traffic Controller is an interactive three-dimensional traffic-management game developed using the Unity game engine and C#. The game simulates a four-way road intersection in which each road contains two directional sections for vehicles travelling in opposite directions. Vehicles follow predefined routes to maintain organized movement through the intersection. The player controls individual vehicles through mouse-based interaction, temporarily pausing a selected vehicle while other vehicles continue moving. The system integrates a 3D environment, vehicle movement, collision detection, a scoring mechanism, game-state management, user-interface elements, background music, and sound effects. Unity Physics and collider components support collision detection, while the Unity UI system displays scores and gameplay controls. The project demonstrates the practical application of 3D scene design, C# scripting, object interaction, physics, and real-time game-state management.**

***Index Terms*—Unity, C#, traffic management, 3D game, vehicle control, collision detection, game development.**

I. Introduction

Traffic congestion and road accidents are common problems in urban areas. Traffic management is important for coordinat-ing vehicles and supporting smoother movement through inter-sections. Traffic Controller is a 3D simulation game designed to represent traffic movement in an interactive environment while providing an entertaining gameplay experience.

The game models a four-way road intersection. Each road contains two directional sections so that vehicles can travel in opposite directions. Predefined routes guide vehicles along their assigned paths and are designed to prevent unwanted crossing or cutting between traffic streams.

The player controls individual vehicles using mouse clicks. When a vehicle is selected, it pauses at its current position and continues to move upon another click. Other vehicles continue moving during this pause, requiring the player to observe the intersection and make timely decisions. A collision ends the current game, while a vehicle that reaches its destination safely contributes to the score.

The project is developed using Unity and C#. The game includes a main menu, pause menu, score counter, collision detection, sound effects, background music, and a game-over screen.

II. Literature Review

Traffic simulation research increasingly uses game engines and external traffic simulators to create interactive three-dimensional environments. These platforms support vehicle visualization, driving behaviour analysis, and the evaluation of traffic scenarios.

Shi, Yang, and Ye developed a Unity–VISSIM co-simulation platform to study interactive driving behaviour. Their work combines Unity-based 3D visualization with traffic simulation, demonstrating the use of interactive virtual environments for studying driving scenarios [1].

Pechinger and Lindner presented Sumonity, an interface connecting the SUMO traffic simulator with Unity. Their approach combines traffic modelling with Unity's graphical and physics capabilities and supports the visualization and control of individual vehicles [2].

Nagy, Májer, and Lepold investigated the integration of Eclipse SUMO, Blender, and Unity 3D to create virtual traffic environments for transportation studies. Their work demonstrates how traffic simulation and 3D modelling can be combined to represent road networks and vehicle movement [3].

Olaverri-Monreal et al. connected the SUMO micro-scopic traffic simulator with Unity 3D to develop a driver-centric simulation platform for evaluating vehicle-to-everything communication-based systems. Their study illustrates the use of a 3D game engine alongside traffic simulation for interactive driving and traffic-safety research [4].

These studies establish the relevance of interactive 3D envi-ronments and vehicle simulation in traffic-related applications. In contrast, Traffic Controller focuses on a simplified four-way intersection in which the player directly selects individual vehicles using mouse clicks. The game combines predefined vehicle routes, temporary vehicle pausing, collision detection, scoring, and user-interface elements to provide an interactive traffic-management experience. Unity documentation provides additional technical guidance for implementing the game's physics, interface, scripting, and audio features [5], [6].

III. Problem Statement

Managing traffic at a busy intersection is challenging because multiple vehicles approach and pass through the central junction at different times. In the game environment, poorly timed control can lead to collisions, interrupt traffic flow, and end the game-play session.

The project addresses this problem by developing a 3D traffic-management game in which four roads meet at a central junction. Each road has two directional sections for vehicles



traveling in opposite directions. Predefined routes are used to keep vehicles within their intended sections. The player selects individual vehicles with mouse clicks, while other vehicles continue to move. The system detects collisions and updates the score according to safe vehicle movement.

The main implementation challenges are maintaining road and lane alignment, keeping vehicles on their assigned routes, selecting the intended vehicle when several are visible, detecting collisions, updating the score and state of the game, and maintaining responsive game-play when multiple vehicles are active.

IV. Objectives

The main objective is to develop an interactive 3D traffic-management game using Unity and C#. The specific objectives are:

- to create a four-road intersection with a central junction and two directional sections on each road;
- to implement predefined vehicle paths using C# scripts;
- to support mouse-based selection and temporary stopping of individual vehicles;
- to allow other vehicles to continue moving while a selected vehicle is paused;
- to detect collisions using Unity Physics;
- to implement a score system based on safe vehicle movement;
- to provide interface elements including the main menu, score display, pause menu, restart option, and game-over screen; and
- to integrate 3D assets, animation, background music, and sound effects.

V. Methodology

The development process follows a sequence of requirement analysis, environment design, vehicle movement implementation, player interaction, collision detection, scoring and interface integration, audio integration, and testing.

A. Requirement Analysis

The requirements include a four-road intersection, two directional sections per road, predefined vehicle movement, mouse-based control, collision detection, scoring, interface states, and audio feedback.

B. Environment Design

In Unity a three-dimensional intersection is created. Road segments, lane sections, vehicles, and environmental assets are positioned to represent the four-way junction.

C. Vehicle Movement

C# scripts control vehicle movement along predefined routes. Each vehicle follows its assigned path and direction to maintain organized movement through the intersection.

D. Mouse-Based Control

Mouse input is used to identify the selected vehicle. Mouse input and a camera ray are used to identify the clicked vehicle and activate control for that vehicle. The selected vehicle pauses in its place and continues to move upon another click, while the remaining vehicles continue to move.

E. Collision Detection and Scoring

Unity colliders and physics components are used to detect contact between vehicles. A collision triggers the game-over state and displays the final score. When a vehicle safely reaches its destination, the score is updated according to the implemented scoring rules.

F. Interface, Audio, and Testing

The Unity UI system provides the main menu, game-play score display, pause menu, resume and restart controls, and game-over screen. Background music and sound effects provide audio feedback. Testing and debugging focus on vehicle movement, route accuracy, selection, collision detection, scoring, interface behaviour, and overall game-play performance. Testing confirmed that vehicles followed their predefined routes, mouse-based selection worked as intended, collisions triggered the game-over state, and score updates and interface controls behaved as expected.

VI. Implementation

The game is implemented in Unity using C#. The scene contains a four-way road intersection, environmental assets, and vehicles configured to follow predefined routes. These routes define the movement paths of vehicles through the intersection. At the beginning of a session, the player selects Start Game from the main menu. The score is initialized, and vehicles begin moving along their assigned routes. Mouse input and camera-ray detection are used to select individual vehicles. A selected vehicle pauses and resumes when clicked again, while other vehicles continue moving.

Unity colliders and physics components detect vehicle collisions. When a collision occurs, the game transitions to the Game Over state and displays the final score. Safe arrival at a destination updates the score according to the implemented scoring rules. The interface supports pausing, resuming, restarting, returning to the main menu, and exiting the session. Background music and sound effects provide audio feedback.

Fig. 1 shows the game control and state-management workflow. Fig. 2 shows the interaction and collision-handling flow.

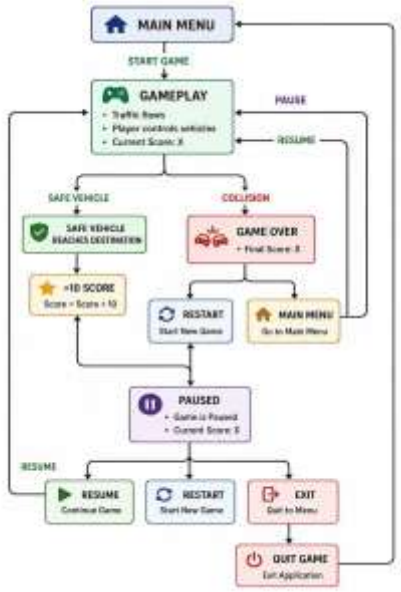


Fig. 1. Game control and state-management workflow.

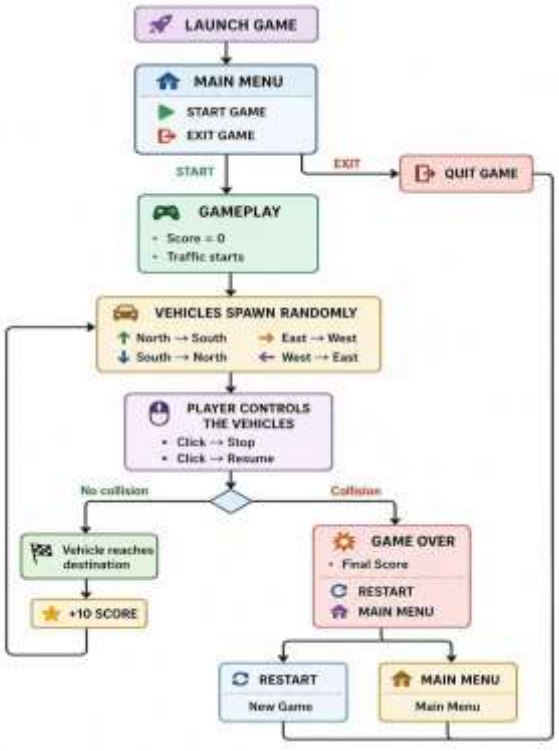


Fig. 2. User interaction and collision-handling workflow.

VII. Gameplay Snapshots

The following screenshots show the implemented interface states. Fig. 3 presents the main menu, Fig. 4 shows active game-play, Fig. 5 presents the pause menu, and Fig. 6 shows the game-over screen and final score.



Fig. 3. Main menu.



Fig. 4. Active game-play with vehicles moving through the intersection.



Fig. 5. Pause menu with resume, restart, and quit options.



Fig. 6. Game-over screen displaying the final score with restart and main menu options.

VIII. CONCLUSION

Traffic Controller is an interactive 3D traffic-management game developed using Unity and C#. It represents vehicle movement at a four-way road intersection using predefined routes, mouse-based vehicle selection, collision detection, scoring, interface states, and audio integration. The project demonstrates practical use of 3D environment design, C# scripting, physics, object interaction, UI design, animation, and audio in real-time gameplay.

The game provides a foundation for further development of interactive traffic simulations. Possible future enhancements include increasing traffic density, introducing intelligent vehicle behaviour, adding multiple intersections and weather conditions, supporting multiplayer gameplay, and implementing smart traffic-signal management.

REFERENCES

[1] X. Shi, S. Yang, and Z. Ye, "Development of a Unity–VISSIM Co-Simulation Platform to Study Interactive Driving Behavior," *Systems*, vol. 11, no. 6, Art. no. 269, 2023, doi: 10.3390/systems11060269.

[2] M. Pechinger and J. Lindner, "Sumonity: Bridging SUMO and Unity for Enhanced Traffic Simulation Experiences," *SUMO Conference Proceedings*, vol. 5, pp. 163–177, 2024, doi: 10.52825/scp.v5i.1115.

[3] V. Nagy, C. J. Májer, and S. Lepold, "Eclipse SUMO and Unity 3D Integration for Emission Studies Based on Driving Behaviour in Virtual Reality Environment," *Procedia Computer Science*, vol. 257, pp. 396–403, 2025, doi: 10.1016/j.procs.2025.03.052.

[4] C. Olaverri-Monreal, J. Errea-Moreno, A. Díaz-Álvarez, C. Biurrun-Quel, L. Serrano-Arriezu, and M. Kuba, "Connection of the SUMO Microscopic Traffic Simulator and the Unity 3D Game Engine to Evaluate V2X Communication-Based Systems," *Sensors*, vol. 18, no. 12, Art. no. 4399, 2018, doi: 10.3390/s18124399.

[5] Unity Technologies, "Unity User Manual." [Online]. Available: https://docs.unity3d.com/Manual/

[6] Unity Technologies, "Audio," Unity User Manual. [Online]. Available: https://docs.unity3d.com/Manual/Audio.html