Journal of Global Research in Electronics and Communication
Volume 1, No. 5May 2025
Available Online at: www.jgrec.info
RESEARCHPAPER

Development of a 2D Car Simulation Game

Nilesh Khemani
Department of Computer Science & Application
Organization Mandsaur University
Mandsaur
nileshkhemani4@gmail.com

***Abstract*—This project is also a simple 2D car racing game developed using Java Swing. The player controls a car with keyboard keys to avoid other cars and collectibles. There are different levels of this game, and as it goes on, they become faster and more difficult. On collecting enough coins, the player moves to the next level. If the car crashes into another car, the game shows a game-over screen and restarts. It shows concepts of game development like animation, collision detection, level progression and user interaction in Java.**

Keywords—Java Swing, 2D Car Game, Keyboard Control, Coin Collection, Collision Detection, Game Levels, Speed Increase, Game Over, Animation, Thread Handling, GUI Development.

I. Introduction

In today's world, digital world, games have now become very essential as they serve as entertainment and learning. The development of 2D games became available for beginners and students with the growth of computer programming and graphical interfaces [1]. The project is a simple 2D car racing game made using Java Swing [2]. It is to offer interaction between an entity (player) through the use of a keyboard, and where the player has a car that can collect coins and avoid other cars. Level progression, increasing difficulty and basic animation are its features [3]. The main objective of the game is to increase the user's engagement by making the game as smooth as possible and coming up with logical challenges, and at the same time, show core programming concepts like event handling, threading, GUI design and also collision detection in Java [4].

II. Objectives

The aim of this project is to develop a simple 2D car racing game utilizing Java Swing such that a user maintains control over a car using keyboard keys to avoid other cars and collect coins [5]. The objective of the game is to make the user have fun while learning to develop event handling, graphic design, collision detection, animation and other important programming skills in java [6]. It also shows how to use games to practice and apply core general programming concepts such as object-oriented programming, control flow and user interface. It has allowed students to improve their coding skills and also helps them to get some 'hands on' experience in game development [7][8]. Thus, students can enhance their skills in problem solving, logical thinking and writing efficient code [9].

III. Tools And Technology

This game is developed by using the Java programming language. Java Swing is chosen for designing graphics and user interfaces [10]. Swing helps to create windows, buttons, and images in the game. The car, coin, and tree images are added using Image Icon. The game controls work with keyboard inputs, and movement is handled using Key Event. The game runs with the help of a loop and Thread Sleep () to control speed [11]. Random positions for coins and cars are set using the Random class.

IV. Game Structures

The 2D car simulation game uses a JFrame as the main window and a JPanel for gameplay, with freely placed elements like cars, trees, and coins. Car movement is controlled by arrow keys, while coins and enemy cars move downward using loops and random positions. The game has three levels with increasing speed, triggered by collecting coins [12]. Collision detection ends the game on enemy impact and increases score on coin collection. A side panel displays speed, coins, and level using JLabels. Here are the game structures are as follows:

- **Main Frame (JFrame):** This is the main game window where everything is shown. The background color and layout are set here.
- **Game Panel (JPanel):** This is where the actual game runs. It includes the player's car, enemy cars, trees, and coins. It uses null layout so that every component can be placed freely using setBounds().
- **Car Movement:** The car is controlled using the left and right arrow keys. Key presses are detected using Key Listener with Key Adapter, and the car's position (x) is updated accordingly.
- **Coin and Enemy Cars:** Coins and enemy cars appear from the top and move down the screen. Their positions are updated in a loop using Thread.sleep() to simulate motion. Coins are generated at random positions using the Random class.
- **Levels and Speed:** The game has 3 levels. With each level, the speed of the game increases. When the player collects 15 coins, the level increases, and the background logic (speed () or highspeed ()) changes the movement speed.
- **Collision Detection:** The game checks for two types of collisions:
  - If the player's car touches an enemy car, an accident happens, and a "Game Over" screen is shown.
  - If the player touches a coin, the coin count increases.
  - Labels and Score Display: The right side of the game window shows speed, current coin count, total coins, and current level using JLabel.

© JGREC 2025, All Rights Reserved

V. GAME FEATURES

The 2D car simulation game includes left/right arrow key controls, random coin generation for scoring, and increasing speed across three levels to raise difficulty. Random cars appear as obstacles collisions trigger a "Game Over" screen, while collecting coins boosts the score. A side panel displays speed, coin count, and current level for real-time feedback. Here are the key features of game are as follows:

- **Car Movement Control-** The player's car can be controlled using the left and right arrow keys. The position of the car is updated accordingly, allowing the player to move the car across the screen.
- **Coins appear randomly on the screen-** The player must collect these coins to increase their score. Each time a coin is collected [13], the score increases.
- **Levels and Speed-** The game has 3 levels. With each level, the game's speed increases, making it more challenging. As the player progresses, the difficulty rises, keeping the game exciting.
- **Random Car-** Random cars move down the screen and can crash into the player's car. If they crash, the game ends, and a 'Game Over' screen shows up. Players need to avoid random cars to keep playing.
- **Collision Detection-** If the player's car collides with an enemy car, the game ends and shows a "Game Over" screen. If the player's car collides with a coin, the coin count increases, and the score is updated.
- **Score Display-** On the right side of the screen, the current speed, the number of coins collected, the total coins, and the current level are displayed using labels.

VI. FUTURE CHANGES

Future improvements include adding JDBC to save scores and progress, mobile and cross-platform support for wider access, more levels with unique challenges, and sound effects and music to enhance player immersion. Here are several future changes are given below:

- **JDBC Integration-** Future versions of the game could integrate JDBC (Java Database Connectivity) to store players' high scores and progress. This would allow players to save their achievements, such as the number of coins collected or the highest level reached, even after they exit the game.
- **Mobile and Cross-Platform Support-** Making the game available on mobile platforms (iOS and Android) or browsers would increase its accessibility and reach a wider audience. Mobile devices, in particular, offer touch controls, which could be an interesting alternative to keyboard-based controls.
- **Level Progression-** Adding more levels, each with unique challenges and themes would provide more variety in gameplay.
- **Sound and Music Integration-** The game lacks sound effects and background music, which could significantly enhance the player's experience. Adding engine sounds, background music, and collision sound effects would create a more immersive experience.

VII. OUTPUTS

The development of a 2D car simulation game. The game involves navigating a car along a tree-lined road, collecting coins, avoiding obstacles, and progressing through multiple levels and an early gameplay screenshot highlights the core mechanics of coin collection and obstacle avoidance. These visuals reflect the game's design focus on progressive difficulty, user engagement, and arcade-style excitement. Here are displays some Figures 1- 3 on 2D game screens are as follows:

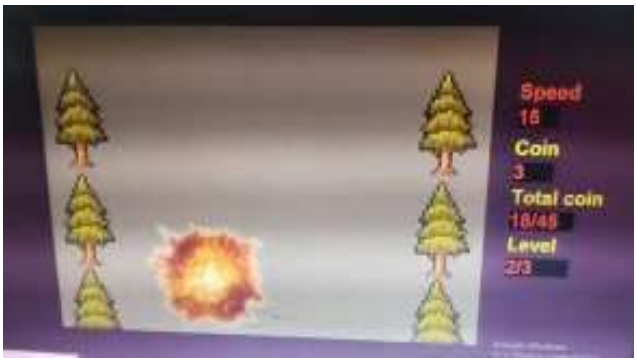

Fig. 1. Explosion scene with speed, coins, and level stats in game

In 2D game, a screenshot from a 2D arcade-style game where the player navigates a path lined with trees. An explosion graphic at the center suggests a collision, possibly ending the current run (shows in Figure 1). On the right, game stats display the player's speed (16), coins collected in the current attempt (3), total coins collected (18/48), and current level (2/3), indicating progressive gameplay with collectibles and increasing difficulty.

Fig. 2. Game over screen from 2D car game

The dramatic "GAME OVER" screen with bold text over a fiery explosion, indicating the end of a session in the development of a 2D car simulation games (see in Figure 2). It emphasizes challenge and consequence typical of arcade-style gameplay.

Fig. 3. 2D car game: coin collection and stats display

The screenshot from a 2D car simulation game where the player navigates a yellow car along a road bordered by trees. An orange car appears ahead, and a coin collectible is placed

© JGREC 2025, All Rights Reserved

in the player's path, (see in Figure 3). On the right side, a game stats panel displays speed (5), coins collected in the current run (2), total coins collected (2/45), and current level (1/3). The scene reflects early gameplay, focused on collecting coins while avoiding obstacles.

VIII. Conclusion

This project successfully demonstrates the development of a simple yet engaging 2D car racing game using Java Swing. It highlights essential programming concepts such as event handling, multithreading, collision detection, and GUI development. The game's structure promotes interactive gameplay by allowing players to control a car, avoid obstacles, and collect coins to progress through increasing levels of difficulty. By implementing game logic with smooth animations and real-time feedback, the project provides an enjoyable user experience while serving as a valuable learning tool for students and beginners in Java programming. Future enhancements like JDBC integration, mobile compatibility, additional levels, and sound effects can significantly improve the game's appeal and functionality. Overall, the project blends entertainment and education, fostering logical thinking, problem-solving, and hands-on coding skills in game development.

References

[1] D. Loiacono, “Learning, Evolution and Adaptation in Racing Games,” in *Proceedings of the 9th Conference on Computing Frontiers*, 2012, pp. 277–284.

[2] A. Balasubramanian, “Improving Legacy Software Quality through AI-Driven Code Smell Detection,” *ESP J. Eng. Technol. Adv.*, vol. 1, no. 1, pp. 245–253, 2021.

[3] J. Roettl and R. Terlutter, “The same video game in 2D, 3D or virtual reality – How does technology impact game evaluation and brand placements?,” *PLoS One*, vol. 13, no. 7, pp. 1–24, 2018, doi: 10.1371/journal.pone.0200724.

[4] A. Chandratreya, S. Dodda, N. Joshi, D. D. Rao, and N. Ramteke, “Robotics and Cobotics: A Comprehensive Review of Technological Advancements, Applications, and Collaborative Robotics in Industry,” *Int. J. Intell. Syst. Appl. Eng.*, vol. 12, no. 21, pp. 1027–1039, 2024.

[5] P. Chatterjee and A. Das, “Enhancing Software Security: A Research-Driven Automation Framework,” *Int. J. Sci. Res. Manag.*, vol. 12, no. 12, pp. 1793–1803, Dec. 2024, doi: 10.18535/ijsrm/v12i12.ec03.

[6] K. Pillai, “Design and Implementation of 2D Game Sprites in Java,” in *35th Annual Conference of The Pennsylvania Association of Computer and Information Science Educators*, 2018.

[7] S. P. Kalava, “Enhancing Software Development with AI-Driven Code Reviews,” *North Am. J. Eng. Res.*, vol. 5, no. 2, pp. 1–7, 2024.

[8] S. V. Inamdar, R. Kumar, and S. Chow, “Method and system for multistage candidate ranking,” 2023

[9] N. V. M. Bindu and S. Singamsetty, “Enhancing Student Engagement and Outcomes through an Innovative Pedagogy for Teaching Big Data Analytics in Undergraduate Level,” *Int. J. Comput. Math. Ideas*, vol. 16, no. 1, pp. 2000–2011, 2024.

[10] K. Logiraj, “Implementation of Java Based Racing Game, Pirate Race, Using Runnable Interface,” *Asian J. Res. Comput. Sci.*, vol. 4, no. 3, pp. 1–8, 2019, doi: 10.9734/ajrcos/2019/v4i330115.

[11] Z. Y. Adam, “A Design and Implementation of A Car Racing Game,” 2024.

[12] S. R. Sagili and T. B. Kinsman, “Drive Dash: Vehicle Crash Insights Reporting System,” in *2024 International Conference on Intelligent Systems and Advanced Applications (ICISAA)*, 2024, pp. 1–6. doi: 10.1109/ICISAA62385.2024.10828724.

[13] S. Tognetti, M. Garbarino, A. T. Bonanno, M. Matteucci, and A. Bonarini, “Enjoyment Recognition from Physiological Data in a Car Racing Game,” *Affin. - Proc. 3rd ACM Work. Affect. Interact. Nat. Environ. Co-located with ACM Multimed. 2010*, pp. 3–8, 2010, doi: 10.1145/1877826.1877830.

© JGREC 2025, All Rights Reserved