"CROSSING THE ROAD" : TURTLE GAME

Chennai, India
av2473@srmist.edu.in

Abinaya Vina
B.Tech CSE IT
SRM University KTR

Abstract - The "Crossing the Road" game, created in Python with the Turtle graphics library, offers an engaging experience where players control a turtle navigating a bustling road. The objective is to safely reach the opposite side, avoiding oncoming traffic. Noteworthy features include player controls utilizing keyboard inputs, a dynamic traffic simulation, and obstacle avoidance that tests players' skills and reflexes. The game incorporates a scoring system, adjustable difficulty levels, and utilizes the Turtle graphics library for visually appealing graphics, and smooth animations. This game not only provides entertainment but also enhances reflexes, decision-making, and hand-eye coordination. Highlighting the Turtle module in Python, it serves as a fun and educational introduction to Python programming, catering to users of all skill levels.

Keywords - "Crossing the Road" game, Python, Turtle graphics library, Obstacle avoidance, Educational gaming

1.INTRODUCTION

The primary goal of this project is to address these issues by developing a Python Turtle-based "Cross-the-Road" game. This game will be designed to entertain users of all ages while also serving as an educational tool for learning and teaching Python programming. By incorporating features such as player control, traffic simulation, scoring, and sound effects, the game aims to engage users in a challenging and interactive experience that promotes skill development and programming literacy. Ultimately, the project seeks to fill the void of accessible, skill-enhancing Python games and promote the use of the Turtle graphics library for game development and educational purposes.

2. PROBLEM STATEMENT

In the realm of recreational and educational Python programming, there exists a compelling need to create an engaging and instructive game that challenges players' agility, reflexes, and strategic thinking. This problem statement identifies the core issues and goals associated with the development of a "Cross-the-Road" game using the Python Turtle graphics library.

3.MOTIVATION

Game development is a skill that can be nurtured from a very young age. As much as game development can be complex, simple game development can also be learned. While learning Python, game development was highlighted and introduced to us. We were keen on creating a simple single-player game using Python. The turtle module caught our attention when we dealt further with game development. We created a game using mainly the turtle module to sharpen our knowledge of the turtle module as well as develop a simple game.

4.LITERATURE SURVEY

S.no	Paper Title	Author	Year	Publisher	Keywords
1	Board Games in the Computer Science Class to Improve Students' Knowledge of the Python Programming Language	Dawid B. Jordaan	2018	IEEE	Well-designed, risk taking
2	Design of King Glory's Game Query System Based on Python	Xiaozhuo Li; Zhenyu Wang	2019	IEEE	Multiplayer, crawler technology, layered architecture
3	Tiny Python Projects: Learn coding and testing with puzzles and games	Ken Youens-Clark	2020	IEEE	-
4	Practical Game Design and Development Pedagogy	Paul J. Diefenbach	2011	IEEE	Game concepts

Literature survey followed by the project

5.OBJECTIVE

The objective of a turtle crossing game is typically for the player to safely guide a turtle from one side of the screen to the other, navigating through various obstacles and challenges along the way. Here are some specific goals and challenges that might be included in such a game:

- Cross the Road: The primary goal is to move the turtle from the bottom of the screen to the top, crossing a busy road filled with moving vehicles.

- Avoid Obstacles: The player must maneuver the turtle to avoid collisions with cars, trucks, or other hazards that move across the screen.

- Timed Crossings: The player may have to reach the other side within a certain time limit, adding urgency to the game.

- Levels of Difficulty: As the game progresses, the speed of vehicles can increase, the number of lanes can multiply, or new types of obstacles can be introduced, making each level more challenging than the last.

6.TOOLS & TECHNIQUES

The following tools and techniques are common in game development, especially in educational and prototype game projects, due the immediate visual feedback they provide. Python Standard Library: The code is written in Python and uses built-in modules such as time for controlling the game's timing.

- Turtle Graphics Module: This is a Python standard library module that provides a basic graphics library that allows users to create pictures and shapes by providing them with a virtual canvas. The turtle module is used here to create the game's graphical elements like the player (turtle), cars (obstacles), and the scoreboard.

- Object-Oriented Programming (OOP): The code is structured around the concept of classes and objects, which is a core part of OOP. This allows for creating reusable code components and a clear structure. The classes Player, CarManager, and Scoreboard are examples of this.

- Game Loop: The while loop in the main.py file is a classic game loop that runs continuously during the game to update the game state, render the graphics, and check for events such as key presses or collisions.

- Event-Driven Programming: The use of screen.listen() and screen.onkey() for binding key presses to the player's movement functions is an example of event-driven programming, where the flow of the program is determined by events such as user actions.

- Collision Detection: The code checks for collisions between the player and cars using the distance() method from the turtle module, which is a fundamental technique in game development for detecting interactions between objects.

- Level Progression: The game includes a simple level progression system where the difficulty increases as the player successfully crosses the screen. This is managed by the Scoreboard class and the CarManager class's level_up method.

- Randomness: The random module is used to introduce randomness into the game, such as the appearance of cars and their positions, which is a common technique to increase the unpredictability and replayability of a game.

- Encapsulation: The use of classes to encapsulate data and methods related to specific game components is a technique used to keep the code modular and manageable.

7.UML DIAGRAM

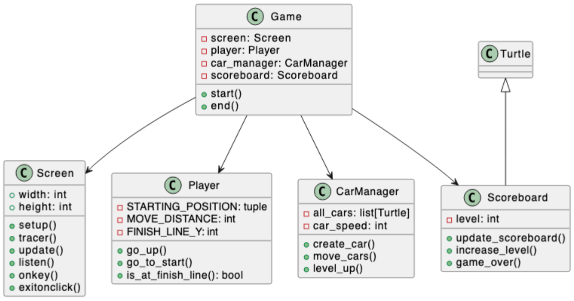
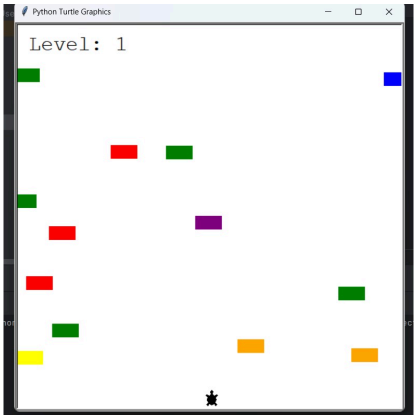
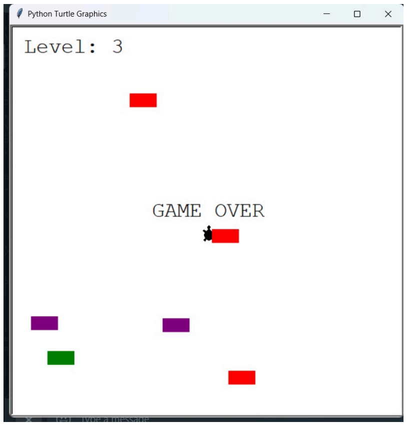


Diagram was made using *MixerBox Diagrams*

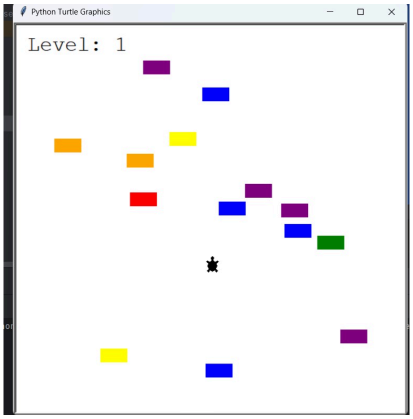
8.OUTPUT





Level: 3

GAME OVER



Level: 1

9.CONCLUSION

The "Crossing the Road" Python Turtle game has been developed with the goal of providing an engaging and educational gaming experience for players of all skill levels. This project has successfully demonstrated the versatility of the Python Turtle graphics library in creating an interactive and visually appealing game.

Throughout the development process, several key objectives were achieved:

- Entertainment and Engagement: The game offers an entertaining experience for players, challenging their reflexes and decision-making skills as they navigate the busy road and avoid collisions with passing cars.

- Educational Value: "Crossing the Road" serves as an excellent learning tool for Python enthusiasts, introducing them to fundamental game development concepts while also encouraging programming literacy and creativity.

- Scalability: The game's adjustable difficulty levels and increasing complexity as players progress make it suitable for a wide range of audiences, from beginners to experienced gamers.

- User-Friendly Design: The intuitive player controls and visual feedback, coupled with sound effects, enhance the overall user experience, making the game accessible and enjoyable.

9.1RESULTS

After thorough development and testing, the "Crossing the Road" game has delivered the following results:

- Functional Gameplay: The game provides a fully functional experience, where players can control the turtle, navigate lanes, and successfully cross the road while avoiding oncoming traffic.

- Scoring System: The scoring mechanism accurately rewards players for successfully crossing the road and penalizes them for collisions, adding a competitive element to the game.

- Graphics and Animation: The Python Turtle graphics library has been effectively used to create visually appealing graphics and smooth animations, enhancing the game's aesthetics.

- Sound Effects: Sound effects, such as car honks and collision sounds, have been integrated to provide audio feedback to players, contributing to a more immersive gaming experience.

In conclusion, the "Crossing the Road" game is a successful project that achieves its objectives of providing both entertainment and educational value. It demonstrates the potential of Python and the Turtle graphics library in game development and serves as a valuable resource for Python learners, educators, and gaming enthusiasts. Whether played for fun or as a learning tool, "Crossing the Road" offers an enjoyable and instructive experience.

10.FUTURE SCOPE

As mentioned previously our interest in game development has only increased after creating our Crossing Turtle Game. Therefore, we would like to enhance our game one day. The following points are our ideas to improve our game.

1. Enhanced Graphics and User Interface:

Custom Graphics: Replace the basic turtle graphics with custom sprites for the player, cars, and environment.

Background: Add a background image to make the game visually appealing.
Animations: Introduce animations for the turtle's movement and car motion to make the game feel more dynamic.

2. Sound Effects and Music:

Sound Effects: Add sound effects for movements, level-ups, and collisions to enhance the user experience.
Background Music: Implement a background score that increases in intensity with each level.

3. Difficulty Levels:

Variable Speeds: Instead of a fixed increment, car speeds could increase based on a curve that gets steeper with each level.
Randomized Car Spawns: Increase the randomness in car generation, with some cars moving faster than others or changing lanes.

4. Player Lives and Health:

Lives System: Give the player a set number of lives before the game ends.
Health Bar: Implement a health bar that decreases with each collision, offering the player more chances before a game is over.

5. Power-Ups and Bonuses:

Power-Ups: Introduce power-ups that can be collected to slow down cars, provide invincibility, or give extra points.
Bonus Scores: Implement bonus scores for quick crossing, consecutive successful crossings without stopping, or collecting special items.

6. Leaderboard and Scoring System:

High Score Tracking: Keep track of high scores locally or online to allow players to compete against each other.
Scoring System: Develop a more complex scoring system based on time taken, number of moves, or close calls with cars.

7. Multiplayer Functionality:

Local Multiplayer: Allow two players to play on the same machine, either cooperatively or competitively.
Online Multiplayer: Implement online multiplayer capabilities for players to compete in real-time.

REFERENCES

- https://docs.python.org/3/library/turtle.html
- https://www.geeksforgeeks.org/turtle-programming-python/
  https://www.javatpoint.com/python-event-driven-programming#:~:text=The%20working%20of%20Event%2DDriven,execute%20and%20in%20which%20order.