Title: System Implementation – Driverless Car

README file

Description

This project implements software to support the operation of a driverless car using OOP principles in Python. It supports three key operations, allowing user interaction through a frontend interface, while the backend simulates data generation and collection. Other required functions, like sensors, cameras, and control panels, are assumed to be imported from other subsystems. The corresponding UML diagrams, modified in response to feedback, are presented in the Appendix for reference.

Key Operations

- **Lane Detection:** Continuously monitors and corrects the car's position within the lane.
- **Obstacle Detection:** Detects obstacles in the path and takes appropriate actions.
- **Emergency Brake:** Activates emergency braking when a critical obstacle is detected.

Module and Class Overview

To better organize the code, classes and functions are defined in separate modules and imported into the main program as needed.

Module	Class	Description

vehicle.py	**Vehicle**	Base class for a generic vehicle with basic functionalities like starting and stopping the engine.
driverless_car.py	**DriverlessCar**	Inherits from Vehicle and includes methods for moving, turning, and braking. Integrates lane detection, obstacle detection, and emergency braking.
lane_detection.py	**LaneDetect**	Handles lane deviation detection and correction.
obstacle_detection.py	**ObstacleDetect**	Manages obstacle detection and initiates slowing down or emergency braking.
emergency_brake.py	**EmergencyBrake**	Activates the emergency brake when needed.
backend.py	**Backend**	Collects and stores the action history.
frontend.py	**Frontend**	Provides a text-based interface for interacting with the car and viewing the history.
test.py		Contains automated tests to validate the functionality of the driverless car system.

Data Structures Used

- **List:** Used in the Backend class to store the history of actions. Lists allow efficient appending, suitable for maintaining a history log.
- **Tuple:** Used to store timestamped entries in the history log in the Backend class. Tuples are immutable and ensure the history entries are not modified after recording

OOP Features Used

- **Inheritance:** The DriverlessCar class inherits from the Vehicle class, allowing it to use and extend the functionalities of the base class.
- **Abstraction:** The system abstracts complex behaviors into simplified interfaces and method calls.
- **Polymorphism:** Methods like move, turn, and brake in the DriverlessCar class exhibit polymorphic behavior, responding differently based on the context.

Execution Instructions

This software is designed to support the operation of a driverless car. It can be executed with additional support from other functions, such as sensors, cameras, and control panels. The Test Module (test.py) can be executed to test the driverless car system and ensure that all components work correctly and interact as expected:

1. Ensure Python 3.x is installed on your IDE or development environment.
2. Ensure all modules are in the same directory.

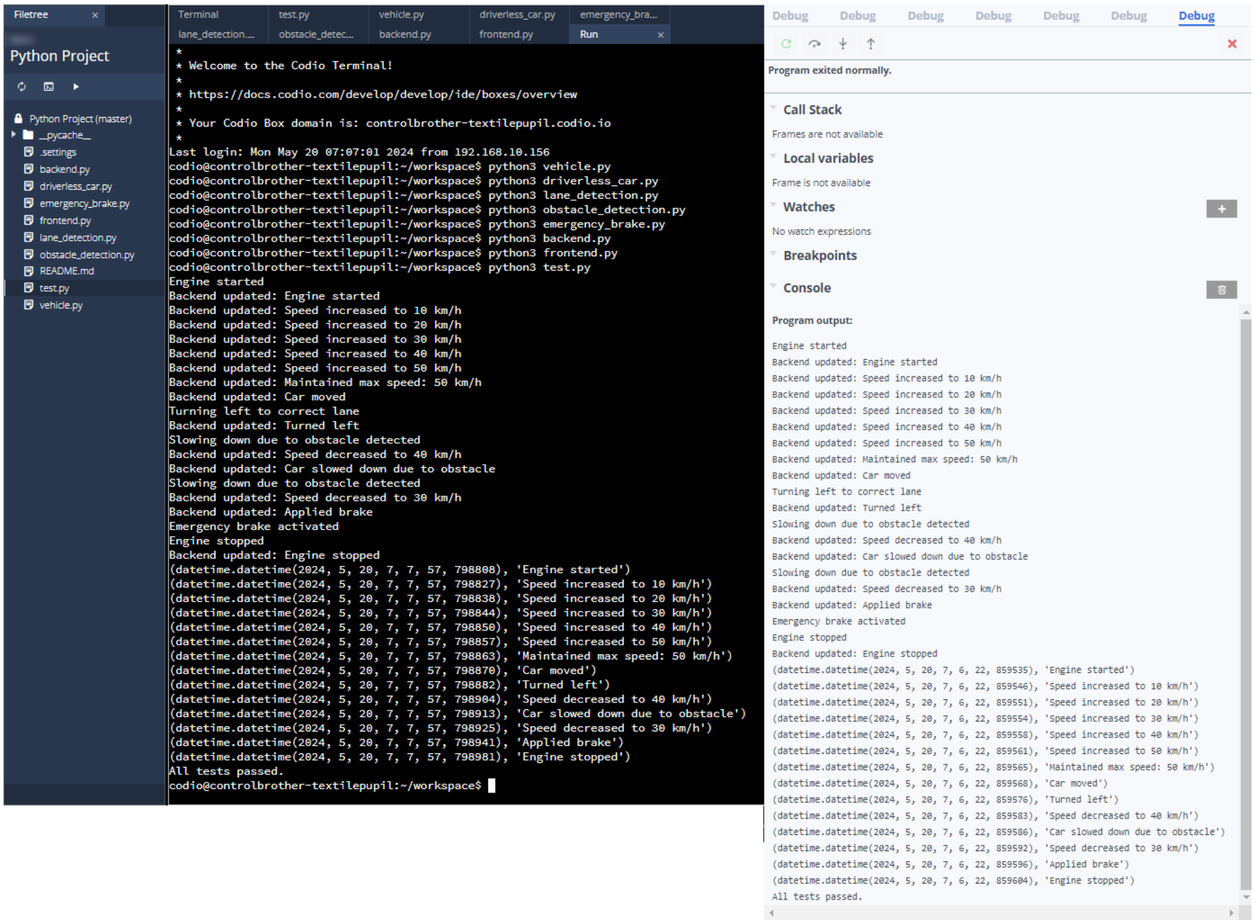
3. Execute the test.py script to perform a series of automated tests validating the system's functionality.

Automated Test Result

The test.py script imports all modules and uses Python’s assert statements to achieve automated testing. The tests simulate various actions of the driverless car to ensure all components interact correctly.

1. **Start the car's engine:** Verifies the engine starts and the speed is initialized to zero.
2. **Move the car forward:** Ensures the car reaches and maintains the maximum speed.
3. **Simulate lane deviation and correction:** Tests the detection and correction of lane deviations.
4. **Simulate obstacle detection and slowing down:** Verifies the car slows down when detecting obstacles at a safe distance.
5. **Simulate further obstacle detection and emergency brake:** Ensures the car applies the emergency brake if the obstacle is too close.
6. **Stop the car:** Verifies the car stops and the engine is turned off.
7. **View history:** Retrieves and displays the history of all actions taken by the car.

Below is the expected output of the test:

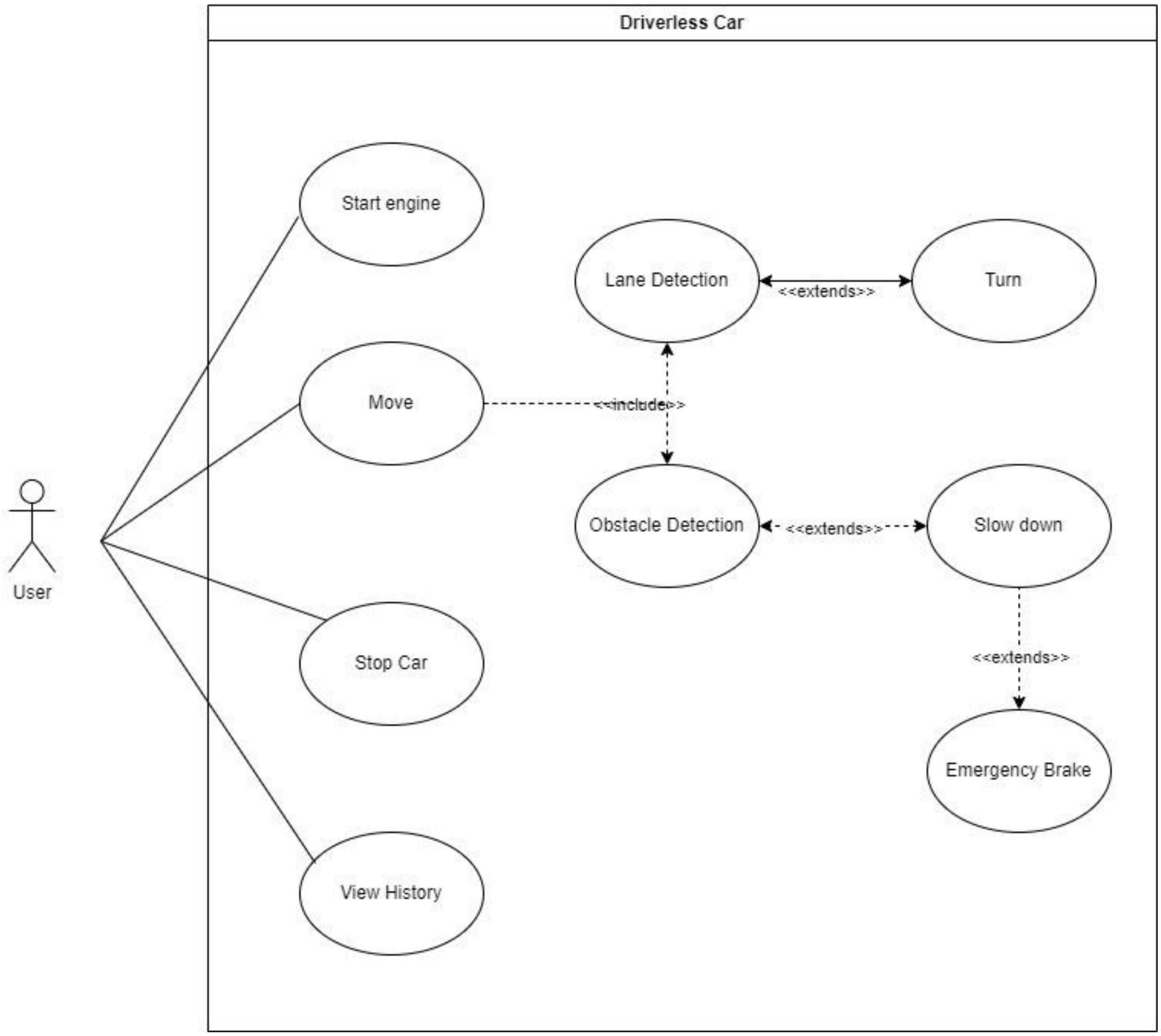


Conclusion

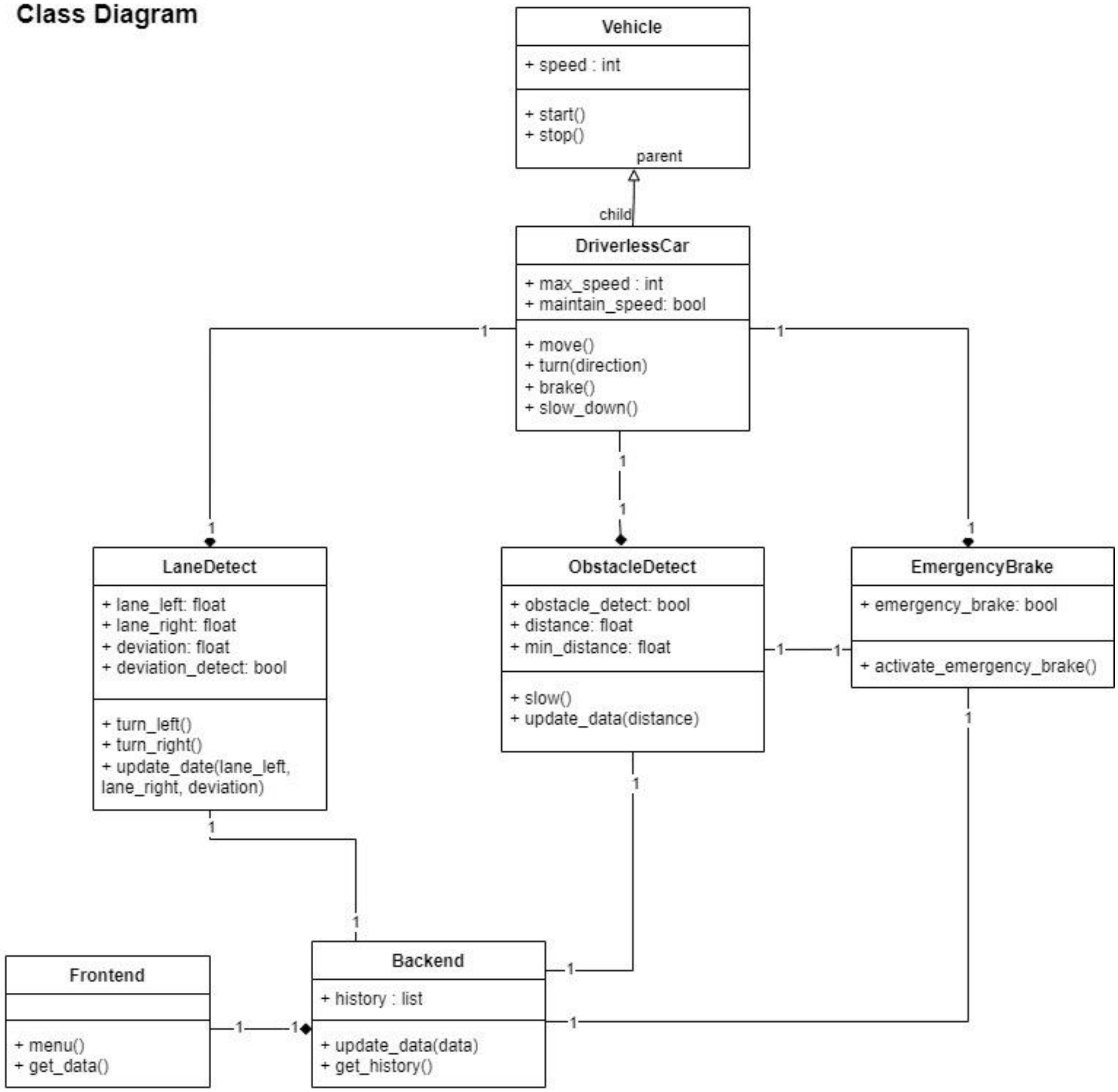
This project demonstrates a implementation of software that supports the operation of a driverless car using OOP principles in Python. The detailed testing and validation processes ensure that each component performs as intended, contributing to the overall reliability and safety of the system. By modularizing the system into distinct components, the project ensures reusability, maintainability, and scalability.

Appendix

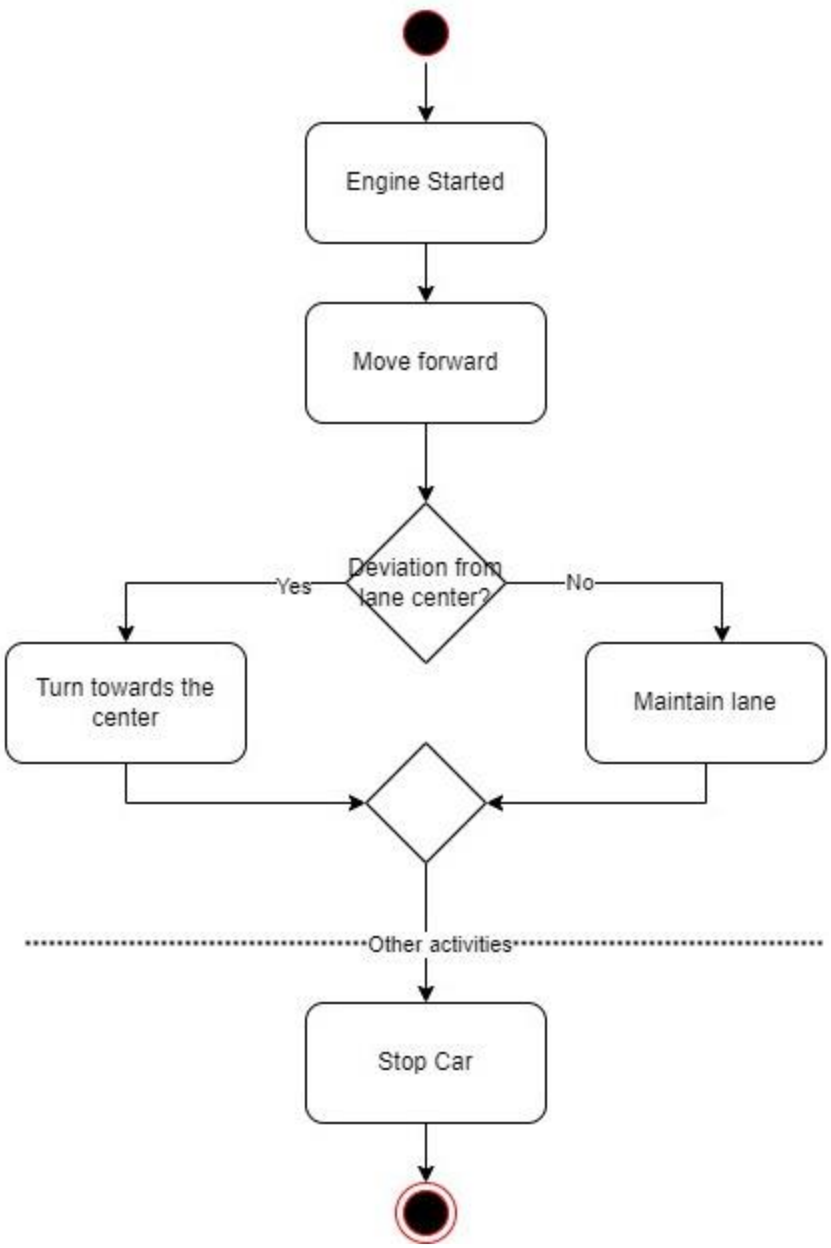
Use Case Diagram



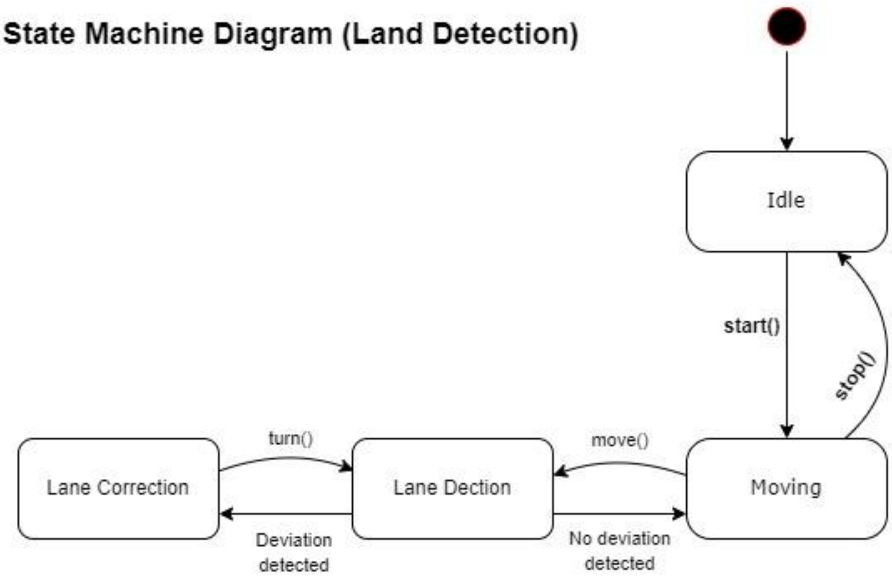
Class Diagram



Activity Diagram (Land Detection)



State Machine Diagram (Land Detection)



Sequence Diagram (Land Detection)