Title: Pathfinding and Traffic Management Simulation for an Ambulance

Brief Description:

This project is a simulation designed to manage picking up a passenger from a set location, pathfinding to the drop off location and the movement of an ambulance in a dynamic environment, incorporating traffic light control and car detection. The simulation uses A* pathfinding to navigate an ambulance through a network of waypoints while responding to traffic lights and avoiding obstacles such as other cars as it transports a passenger from it's pickup location to the hospital. The system integrates real-time decision-making to ensure realistic and responsive vehicle behavior in a traffic scenario.

Environment Description:

The environment consists of a network of waypoints connected by edges that represent possible paths for the ambulance. The simulation includes various elements such as traffic lights, other vehicles (cars), and specific locations for picking up and dropping off the passenger. The traffic lights change colors to simulate real-world traffic conditions, and cars act as dynamic obstacles that the ambulance must avoid.

Number of Characters and Their Functionality:

1. **Ambulance (Intelligent Agent)**:
    - **Functionality**: The ambulance is the primary intelligent agent that navigates through the environment. It uses A* pathfinding to determine the optimal route from its current position to a series of goals (start point, pickup location, drop-off location, and end point). The ambulance also interacts with traffic lights and other cars, stopping when necessary and resuming movement when conditions permit.
2. **Passenger**:
    - **Functionality**: Represents the game object being picked up and dropped off by the ambulance. It does not interact with the environment but is crucial for testing the ambulance's ability to pick up and drop off passengers at designated locations. It has the idle animator applied to it to simulate a passenger waiting to be picked up. The Passenger, when picked up becomes a child of the ambulance and then when dropped off becomes its own game object again. The method is used to replicate the passenger being within the ambulance as it is taken to its location.
3. **Traffic Lights**:

- **Functionality**: Control the flow of the ambulance by changing states (red, yellow, green). The ambulance must stop when the light is red and proceed when it turns green.

4. **Cars**:
    - **Functionality**: Act as moving obstacles that the ambulance must avoid. Their presence influences the ambulance's decision-making regarding movement.

Intelligent Agent Implementation (A Pathfinding):*

The intelligent agent (ambulance) is implemented using the A* pathfinding algorithm, which calculates the shortest path between two points (nodes) in a graph. The pathfinding system is managed by:

- **AStarManager**: Handles pathfinding queries and manages connections between waypoints.
- **VisGraphWaypointManager**: Provides the connections between waypoints, which are used to build the graph for A*.

Types of Obstacles:

5. **Traffic Lights**: Static obstacles that control the movement of the ambulance based on their state.
6. **Cars**: Dynamic obstacles that the ambulance must detect and avoid. Their presence is managed through collision detection.

Start and Goal Points:

- **Start Point**: The initial location of the ambulance from where it begins its journey.
- **Pickup Location**: The point where the ambulance must pick up the passenger.
- **Drop-off Location**: The point where the ambulance must drop off the passenger.
- **End Point**: The destination of the ambulance after dropping off the passenger.

User Input Values:

- **The** input values primarily involve configuring the starting, pick-up, drop off, and ending points, traffic light states, and the presence of cars in the simulation.
- **Frequency**: Inputs are set up during initialisation and are adjusted as needed based on simulation requirements (e.g., changing traffic light states or car positions).

Assets Used:

- **Pathfinding Assets**: Custom scripts for A* pathfinding and waypoint management.
- Polygon City Pack asset
- Basic Motions Assest
- Starter Assest
- SynthStudios Assest

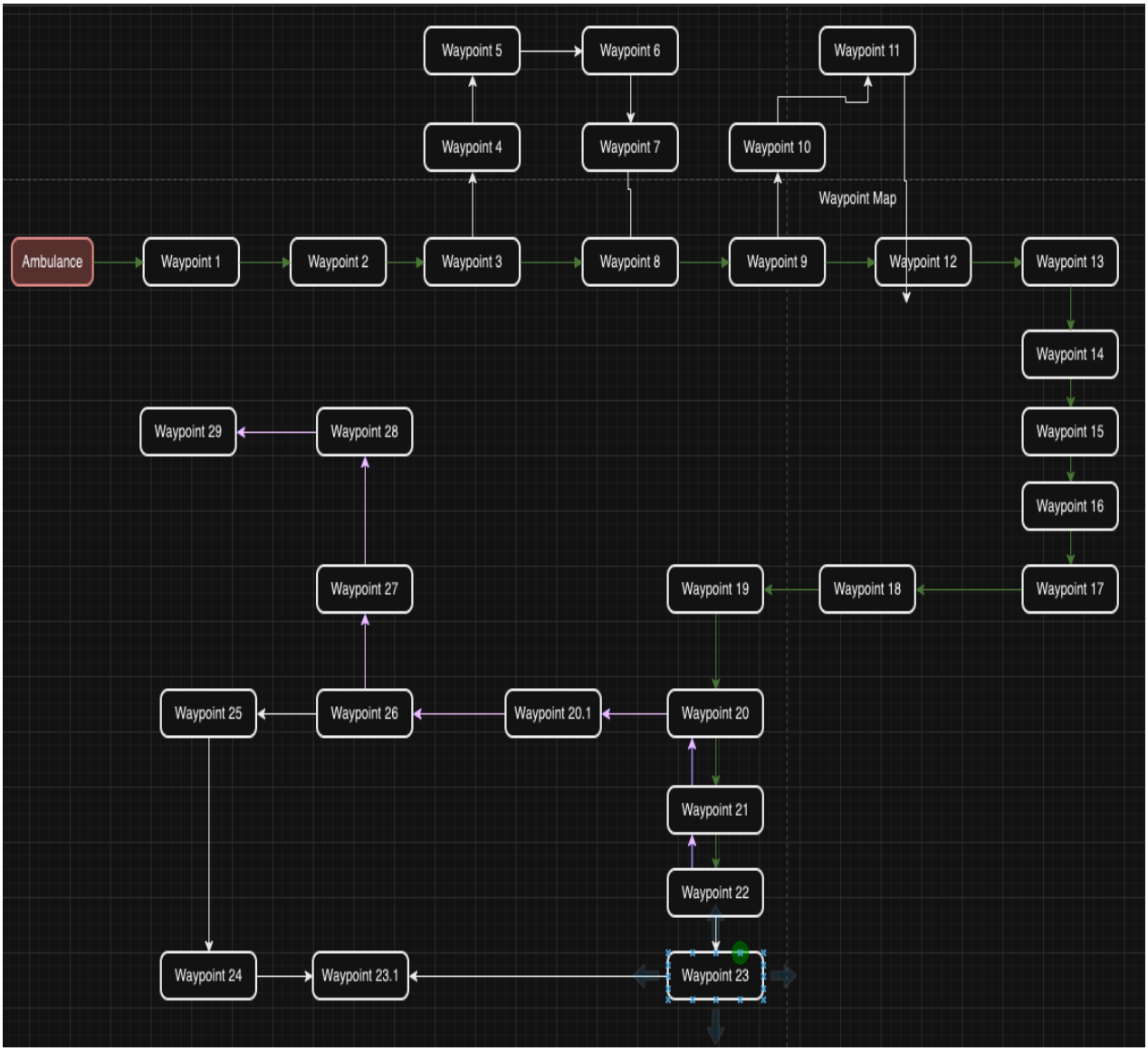
Unity Version:

- Developed in Unity version 2023.1.0 or later.

Graphical Representation of Waypoint Graph:

The waypoint graph consists of vertices (waypoints) and edges (connections between waypoints). The graphical representation can be visualised as follows:

- **Vertices**: Represented as points in space, each corresponding to a waypoint.
- **Edges**: Represented as lines connecting the vertices, indicating possible paths between waypoints.
- The green arrows represent the path taken to pick up and then drop off the passenger.
- The pink arrows represent the path taken to the End destination.
- The white arrows represent the paths that were not taken.



In this simple representation:

- **Waypoint 01 to Waypoint 29** are vertices.
- **Edges** represent the connections that the A* algorithm uses to navigate between waypoints.

This summary captures the essence of the project, detailing the functionality of different components, the environment setup, and the implementation of the pathfinding system.