Alex Delgado Linde
Inteligencia Artificial - CITM

Final Project: Interactive Simulation with AI Techniques

Objective

Develop an interactive Unity scene that integrates multiple AI techniques, showcasing autonomous agents in a dynamic and creative environment. This report details the implementation, design decisions, and challenges faced during the development of the project.

Scene Overview

The scene is set in a bustling city that combines dynamic elements such as pedestrians, cars, and interactive zones. AI agents, including a hero, villains, police, and a criminal, operate across two NavMesh surfaces: one for vehicles on the road and another for pedestrians and rooftop interactions. Autonomous behaviors and coordination ensure a lively, interactive environment.

AI Techniques Implemented

1. Pathfinding and Navigation

Agents Utilizing Pathfinding:

The implementation of pathfinding and navigation evolved iteratively, introducing new agents and refining their behavior as the scene expanded:

1. **Pedestrians (Wandering Agents):**
    - **Initial Setup:**
        - The first implementation involved pedestrians using NavMeshAgents to wander randomly within a specified radius. This created a dynamic city-like movement.
        - A Crosswalk Manager was added to toggle pedestrian movement at crosswalks based on traffic light states.
    - **Improvements:**
        - Pedestrian navigation was refined to avoid static obstacles, such as buildings and crosswalks.
        - Coordination with vehicles at crosswalks was introduced, enabling safe navigation.
2. **Delivery Guy (Chasing Agent):**

- **Initial Implementation:**
    - The Delivery Guy was added to simulate a chasing agent. Initially, it moved between pedestrians in a predefined sequence.
- **Refinements:**
    - Dynamic chasing behavior was implemented, where the agent selects the next pedestrian randomly.
    - Smooth acceleration and deceleration mechanics were introduced, especially at crosswalks.
    - Pathfinding was optimized for efficiency, ensuring the agent adapts to changes in the environment, such as new pedestrian positions.

3. **Cars:**
    - **Initial Setup:**
        - Cars were implemented with basic waypoint-based navigation using a road-specific NavMesh surface.
        - This enabled vehicles to follow predefined paths on roads.
    - **Enhancements:**
        - Smooth transitions between waypoints were added using blending techniques, resulting in realistic turning.
        - Interaction with crosswalks was refined, allowing cars to stop or slow down based on pedestrian activity.
        - Collision avoidance mechanics ensured vehicles navigate without disrupting the flow of the scene.
4. **Rooftop Navigation (Hero and Villains):**
    - **Initial Implementation:**
        - Hero and villain agents were introduced with the ability to navigate rooftops using offmesh links.
        - This allowed agents to jump between buildings, adding verticality to the navigation.
    - **Improvements:**
        - Villains were equipped with frustum vision to detect the hero dynamically.
        - Navigation behaviors were coordinated between agents, enabling group pursuits.

Implementation Notes:

- Two distinct NavMesh surfaces were employed:
    - **Roads:** Dedicated to cars for waypoint-based navigation.
    - **Pedestrians and Rooftops:** Used for all other agents, including pedestrians, the hero, and villains.
- OffMesh Links added seamless transitions for rooftop navigation, enhancing realism and interactivity.
- The Crosswalk Manager acted as a central hub, ensuring coordination between vehicles and pedestrians, dynamically toggling walkable areas.
-

2. Autonomous Movements

Agents Demonstrating Autonomous Behaviors:

1. **Bird Flocking Group:**
   - Inspired by Sebastian Lague's tutorial on flocking, I implemented the Boids algorithm to simulate realistic flocking behavior among a group of birds. This involved three key behaviors:
     - **Alignment:** Each bird steers itself to align with the average heading of its nearby flockmates.
     - **Cohesion:** Birds move toward the average position of their neighbors, maintaining group cohesion.
     - **Separation:** Birds avoid crowding by steering away from nearby flockmates, ensuring collision-free movement.
   - Compute shaders were utilized to handle the complex calculations required for these behaviors, enabling efficient parallel processing and ensuring smooth performance even with a large number of agents.
   - Collision detection was achieved through raycasting in multiple directions, allowing birds to dynamically avoid obstacles in their environment.
2. **Delivery Guy:**
   - Executes chasing behavior by dynamically moving between pedestrians, simulating the behavior of an ice cream vendor in a busy city.
   - Smooth acceleration and deceleration transitions were implemented using coroutines, ensuring fluid and natural movements.
   - Navigation paths dynamically adapt to environmental changes, such as shifts in pedestrian positions or new obstacles, enhancing the realism of the agent's behavior.

Agents Demonstrating Autonomous Behaviors:

1. **Bird Flocking Group:**
   - Compute shaders were chosen to implement flocking behavior because they enable efficient parallel processing of large numbers of agents (boids). Unlike traditional CPU-based approaches, compute shaders utilize the GPU to handle the complex calculations required for alignment, cohesion, and separation rules simultaneously across all agents. This significantly improves performance, ensuring smooth movement and responsiveness even in scenes with high agent density. Additionally, this method minimizes frame drops and maintains high simulation fidelity, crucial for dynamic environments like the one in this project.
   - Raycasting was implemented by equipping each bird agent with multiple rays cast in different directions, simulating a field of vision to detect potential obstacles. The detection range and number of rays were carefully tuned to balance performance and accuracy. A smaller number of rays with strategically chosen angles ensured that collision avoidance remained effective without overwhelming the computation, especially in dense flocking scenarios. Upon detecting an obstacle, agents adjusted their trajectory dynamically, ensuring smooth and responsive movement.

- Demonstrates the three fundamental boid behaviors: alignment, cohesion, and separation.

2. **Delivery Guy:**
    - Executes chasing behavior by moving between pedestrians dynamically.
    - Smooth transitions in acceleration and deceleration ensure fluid movement.
    -

---

3. Formation Motion

Development Process and Chronology:

The concept of formation motion was progressively introduced and refined during the development of the scene. This involved creating agents that could work together cohesively to achieve their goals, both in group behaviors and inter-agent communication:

1. **Villains (Zombie-like Agents):**
    - **Initial Implementation:**
        - Villains were designed to act as a coordinated group with the ability to detect and pursue the hero using frustum-based vision.
        - Initial group behavior focused on individual agents reacting to the hero's position independently.
    - **Improvements:**
        - A world interfacing mechanism was introduced to allow villains to communicate when one detected the hero. This enabled coordinated pursuit, where nearby villains adjusted their behavior based on shared information.
        - Group dynamics were enhanced by ensuring villains maintained spatial awareness, avoiding collisions while pursuing the hero in a cooperative manner.
2. **Crosswalk Coordination:**
    - **Initial Setup:**
        - Crosswalks were first implemented as static zones with walkable states toggled based on traffic light conditions.
    - **Enhancements:**
        - Crosswalks evolved to serve as hubs for communication between vehicles and pedestrians. The Crosswalk Manager dynamically adjusted the walkable state of the crosswalk based on light indicators, ensuring safe transitions for both pedestrians and vehicles.
        - Visual feedback was added through light indicators (green for vehicles and red for pedestrians, or vice versa), providing a clear representation of crosswalk states.
        - This coordination prevented collisions and maintained a realistic flow of movement in the city.

Key Features:

1. **Villain Group Dynamics:**

    - Frustum vision for detection and pursuit.
    - Communication between agents for coordinated responses.
2. **Crosswalk Communication:**
    - Dynamic toggling of walkable states.
    - Visual feedback with light indicators for real-time state representation.
    - Coordination between vehicles and pedestrians to prevent collisions.

Impact on Scene Design:

The inclusion of formation motion added a layer of complexity and realism to the scene. Villains demonstrated cooperative behavior, creating dynamic challenges for the hero, while crosswalks ensured seamless integration between vehicles and pedestrians, enhancing the overall flow of the city environment.

Agents Utilizing Coordinated Group Behavior:

1. **Villains (Zombie-like Agents):**
    - Use world interfacing to communicate and coordinate when detecting their target (the hero).
    - Group behavior leverages frustum-based vision to track and pursue the hero effectively.
2. **Crosswalk Coordination:**
    - Crosswalks act as communication hubs for pedestrians and vehicles.
    - Dynamic state changes (e.g., walkable or not) are visually represented with light indicators.
    - Ensures coordinated flow between agents, preventing collisions.
    -

---

4. Decision Making

Development Process and Chronology:

The decision-making logic was iteratively developed and enhanced to create dynamic and interactive behaviors for the agents. This process involved implementing finite state machines (FSMs) and adding inter-agent communication for coordinated responses:

1. **Hero:**
    - **Initial Implementation:**
        - The hero agent was programmed to patrol predefined waypoints using a basic state machine.
        - Early functionality included transitioning between patrol and evasion states based on the proximity of villains.
    - **Improvements:**
        - FSM was expanded to include additional states, such as idle and pursuit, depending on environmental triggers.

        - The hero’s interactions with villains were refined to incorporate dynamic reactions, leveraging offmesh links for rooftop navigation and escape routes.
2. **Villains:**
    - **Initial Setup:**
        - Villains were introduced with a simple chase state, where they pursued the hero upon detection.
    - **Enhancements:**
        - A multi-state FSM was implemented, including states such as idle, patrol, and chase.
        - Frustum-based vision allowed villains to dynamically detect the hero and coordinate with other villains nearby.
        - Communication between villains enabled group-level decision-making, ensuring coordinated pursuit while avoiding collisions.
3. **Crosswalk Manager:**
    - While primarily a coordination tool, the Crosswalk Manager also demonstrated decision-making by dynamically toggling states based on traffic light cycles. This ensured vehicles and pedestrians moved safely without collisions.

Key Features:

1. **Hero FSM:**
    - Transitioned between patrol, evade, and idle states based on villain proximity and interactions.
    - Leveraged offmesh links for advanced navigation in rooftop chases.
2. **Villain FSM and Communication:**
    - Enabled cooperative chasing behaviors by sharing detection information among agents.
    - Avoided redundant pursuit paths, optimizing group behavior and increasing the challenge for the hero.
3. **Dynamic Coordination:**
    - Crosswalk state changes influenced pedestrian and vehicle behaviors, demonstrating environmental decision-making at a broader scale.

Impact on Scene Design:

The decision-making logic added depth to agent behaviors, making interactions more dynamic and realistic. The hero’s evasive maneuvers and the villains’ coordinated chases created engaging rooftop dynamics, while the Crosswalk Manager ensured seamless integration between vehicles and pedestrians.

Decision-Making Agents:

1. **Hero:**
    - Operates with a finite state machine (FSM) to switch between states (e.g., patrolling, evading villains).
    - Patrols waypoints while dynamically reacting to villains’ pursuit.
2. **Villains:**

- Utilize FSMs to handle their states (idle, chasing, coordinating with other villains).
- Communicate dynamically with each other when detecting the hero, enhancing group coordination.

---

5. Machine Learning (ML-Agents)

Development Process and Chronology:

Machine learning was implemented to create intelligent and adaptive behaviors for the police and criminal agents. The process leveraged Unity's ML-Agents framework to train agents in dynamic and challenging scenarios:

1. **Police Agent:**
    - **Initial Implementation:**
        - The police agent was tasked with pursuing the criminal efficiently.
        - Early training sessions involved simple pursuit scenarios without obstacles, focusing on learning to reduce the distance to the criminal.
    - **Improvements:**
        - Training environments were expanded to include cars as dynamic obstacles.
        - The police agent learned to avoid collisions with cars by penalizing such events, encouraging safer navigation.
        - Observations included the criminal's position, velocity, and the positions of nearby cars, allowing for more informed decision-making.
2. **Criminal Agent:**
    - **Initial Setup:**
        - The criminal agent was trained to maximize the distance from the police.
        - Early scenarios focused on basic evasion, with the agent moving away from the police.
    - **Enhancements:**
        - Complex environments were introduced, including hiding spots and obstacles like cars.
        - Observations were expanded to include the positions of police, cars, and potential escape routes.
        - Rewards were structured to encourage strategic evasion, such as utilizing obstacles or maintaining a safe distance over time.

Key Features of ML-Agents:

1. **Police:**
    - Learned efficient pursuit strategies while avoiding collisions.
    - Demonstrated adaptability to dynamic environments with moving obstacles.
2. **Criminal:**
    - Exhibited intelligent evasion behaviors, leveraging environmental features to escape.

- Adapted to pursue long-term safety by avoiding predictable paths.

Training Setup:

- The agents were trained using a reward and penalty system:
  - **Police Rewards:** Closing the distance to the criminal.
  - **Police Penalties:** Collisions with cars or failing to catch the criminal within a time limit.
  - **Criminal Rewards:** Increasing distance from the police and maintaining separation over time.
  - **Criminal Penalties:** Collisions with cars or being caught by the police.
- Training environments included varied layouts to improve generalization and adaptability.

Impact on Scene Design:

The ML-Agents integration added an advanced layer of interactivity to the scene. The police and criminal agents created dynamic and engaging scenarios, reacting intelligently to each other and their environment. Their behavior added depth to the city simulation, making it more immersive and unpredictable.

ML-Agents Utilized:

1. **Police:**
   - Trained to pursue the criminal efficiently.
   - Observes environmental factors such as the criminal's position and nearby vehicles.
   - Penalized for colliding with cars, encouraging safe navigation.
2. **Criminal:**
   - Trained to evade the police by hiding and avoiding detection.
   - Observes police position, nearby vehicles, and potential hiding spots.
   - Receives rewards for increasing distance from the police and penalties for unsafe behavior.

Scene Design

- **Environment:**
  - A lively city with roads, sidewalks, and interconnected rooftops.
  - Crosswalks equipped with dynamic lights coordinate vehicle and pedestrian flow.
- **Interactions:**
  - Hero and villains engage in rooftop chases, leveraging offmesh links for navigation.
  - Pedestrians and vehicles interact at crosswalks, creating dynamic flow.
  - Police and criminal agents navigate the city, dynamically reacting to obstacles and each other.

Challenges and Solutions

1. **Smooth Movement for Cars:**
   - Implemented waypoint blending with `Vector3.Lerp` and smooth deceleration using coroutines.
2. **Complex Flocking Dynamics:**
   - Optimized boid calculations with compute shaders for high performance.
3. **Coordinated Group Behavior:**
   - Leveraged world interfacing and frustum vision for villain communication.
4. **Crosswalk Coordination:**
   - Designed a central manager to handle dynamic states and prevent collisions.
5. **ML-Agent Training:**
   - Balanced reward structures to encourage desired behaviors for police and criminal agents.
