CSC 481/581 Fall 2025 - Milestone 3

Overview

Your task for this assignment is to expand the functionality of your game engine, implement a game to demonstrate its features, and write a detailed report on your design and results. You will submit your code in two parts (team and individual) and an individual writeup.

As always, you are expected to abide by the University's Academic Integrity Policy, which includes providing appropriate attribution for all external sources of information consulted.

Point Distribution

- **Total Points:** 125
- **CSC 481 Students:** Required to complete 100 points (Parts 1 & 2). Scores above 100 will be capped at 100. Part 3 is optional.
- **CSC 581 Students:** Required to complete all 125 points.
- **Writeup:** The individual writeup (Part 4) constitutes 50% of the total grade for this assignment.

Part 1: Team Engine Component (25 Points Total)

This component represents the work your team will complete and submit together. It includes the core engine architecture for the game object model, networking, and performance testing.

A. Game Object Model (10 points)

1. **Design and Implement an Object Model:** Design and implement a runtime game object model. This model must be component-based, property-based, or a custom design of your own. You may **not** use a monolithic inheritance hierarchy.
2. **Ensure Extensibility:** The model should be flexible enough to reasonably implement new types of game objects in the future.

B. Multithreaded, Networked Scene (15 points)

1. **Integrate Networking:** Integrate your game object model with your multithreaded network code (either client-server or peer-to-peer).
2. **Handle Connections:** The network system must be multithreaded, reading and sending data concurrently while accepting an arbitrary number of incoming connections.
3. **Propagate Player Movement:** Ensure that keyboard inputs controlling a unique player character on each client are propagated across the network and rendered on all other clients with minimal latency (<100 ms is optimal).

4. **Synchronize Scene:** Draw the complete game scene on up to four simultaneous clients, including the movement of all player-controlled objects and dynamic platforms.
5. **Utilize Object Model:** Your runtime object model must be used on every network endpoint (all clients and the server, if applicable).
6. **Manage Disconnects:** Handle client disconnects gracefully. When a client disconnects, their player object should be removed from the scene, and all remaining clients (and the server) must continue to operate normally.

C. Performance Comparison Framework (12.5 points - *Required for 581, Optional for 481*)

1. **Implement a Second Data Format:** Implement a second, distinct strategy for exchanging data across the network using 0MQ. Your goal is to create two approaches that are different enough to produce a noticeable performance difference without altering the core gameplay experience. Examples include:
    - Varying the type of information sent (e.g., sending full object states vs. sending only input deltas).
    - Varying the data representation (e.g., sending serialized objects vs. sending raw byte arrays).
    - Comparing the performance of a client-server model vs. a peer-to-peer model.
2. **Conduct Performance Experiments:** Design and run a set of at least three different experiments to measure and compare the performance of your two networking strategies.
3. **Vary Experimental Conditions:** For each experiment, vary conditions such as the number of clients connected and the number of static and moving objects in the scene (e.g., test with 10, 100, and 500 moving objects).
4. **Measure and Record Data:** Measure the time required to complete a set number of game loop iterations (e.g., 100,000) under each condition. Run each experiment multiple times (at least 5 is recommended) and record the average and variance.

Part 2: Individual Game Component (25 Points)

Using your team's engine, each student must individually implement and submit a 2D game that demonstrates the engine's capabilities. A platformer is a good starting point. The game must include at least the following objects and mechanics:

1. **Player Characters:** Implement characters that are controlled by the client. Characters must respond to keyboard inputs such as to move left, move right, and jump in the case of a platformer.
2. **Static Platforms:** Create static platforms with a position, size, and color/texture. You must demonstrate at least three different platforms, each with a unique combination of these properties.
3. **Moving Platforms:** Implement platforms (or other entities) that move in a patterned way (e.g., horizontally, vertically, or in a circular path). You must demonstrate at least two different movement patterns.

4. **Spawn Points:** Create hidden (non-rendered) objects using your object model that mark starting locations for player characters. Your environment must contain multiple spawn points.
5. **Death Zones:** Implement hidden objects that act as boundaries. Use collision detection to identify when a character enters a death zone. Upon collision, the character must be teleported to a designated spawn point. Your game must have at least one functional death zone.
6. **Side-Scrolling Boundaries:** Implement a hidden boundary object that, when a character collides with it, triggers a lateral translation of all objects in the scene to create a side-scrolling effect. This will cause some objects to move into view while others move out of view. Alternatively, you may construct a camera which translates the display over an abstract internal space representation.

Part 3: Individual Writeup Component (50 points for 481, 62.5 for 581)

Each student must submit a 3-4 page paper summarizing their work. The paper should be single-spaced and a minimum of three full pages. It is **strongly** suggested that you go beyond simply answering the questions and think creatively about your design, its implications, and your experience. This writeup accounts for **50% of your total grade**.

Your writeup must address the following:

1. **Engine Design:**
   - Motivate and defend your team's game object model design. What were the alternatives, and why did you choose this approach?
   - What does your design enable for future development? What are its limitations?
   - What behaviors, properties, or components were necessary to implement the required game objects?
2. **Network Architecture:**
   - Describe your team's decision for the networking model (client-server, peer-to-peer, or both).
   - Detail the data format used to send information across the network.
3. **Performance Analysis (*581 students must include this*):**
   - Describe the two data exchange formats you implemented and compared.
   - Present the data collected from your experiments, explaining your methodology.
   - Analyze the results. Which approach was more performant and under what conditions? What conclusions can you draw from the data?
4. **Reflection:**
   - What aspects of the design and implementation worked as you expected, and what did not?
   - What were the most significant challenges your team faced, and how did you overcome them?
5. **Appendix (Does not count toward the page requirement):**
   - Include relevant screenshots of your individual game in action. Reference these screenshots within your paper to support your points.

    - Include graphs and visualizations from your performance comparison experiments (Part 1C). Reference these by number in the main body of your paper.

Submission Checklist

You have three separate deliverables for this assignment:

1. **Team Submission:** A single submission for your team containing the complete source code for the game engine.
2. **Individual Game Submission:** Your individual submission containing the game you built using the team's engine.
3. **Individual Writeup Submission:** Your individual submission of the 3-4 page paper with its appendix.