Dept. of Computer Science & Software Eng., Concordia University
COMP 476/6331 --- Winter 2018

Advanced Game Development

Assignment 3 version 2 --- Due April 2, 2018

PURPOSE: This assignment will give you the opportunity to learn more about programming a networked multiplayer game. This assignment contains both **theoretical** questions, which you don't need to implement, and a **practical** question, which requires you to write a C# program for Unity.

SUBMISSION: The assignments must be done individually. On-line submission is required (see details at end) – a hard copy will not be read or evaluated.

PREPARATION: Parts of this assignment require knowledge of basic concepts from parts of the material covered in the course slides (exact sections of which books the material is based on are listed on the course schedule).

Question #1: (12%) [Theoretical Question]

Consider using a Hierarchical N-gram predictor to predict the next move of your opponent for a fighting game where the only actions are L and R moves. Suppose that we have the following training data (observed sequence of moves):

L R R R L R R L L R L R L R L R R L R R L R R R R L L R R L R R

a) (5%) Using a hierarchical 3-gram predictor, what is the predicted next action for input "L R R R", if we want at least 5 samples for prediction?
b) (4%) Using a hierarchical 3-gram predictor, what is the predicted next action for input "L R R R", if we want at least 15 samples for prediction?
c) (3%) Using a hierarchical 3-gram predictor, what is the predicted next action for input "L R R R", if we want at least 30 samples for prediction?

For each question above, give the details of your prediction. Just writing "R" or "L" without any justification will result in no marks.

Question #2: (8%) [Theoretical Question]

Consider a sphere with radius 3 and center point (1, 1, 1) colliding with the triangular face of an object. The vertices of the triangle are (-1, 1, -3), (4, 0, -1), and (-2, -2, -2). Determine the following:

a) (5%) the contact normal; and
b) (3%) the interpenetration depth.

Question #3: (80%) [Practical Question]

Problem Statement:

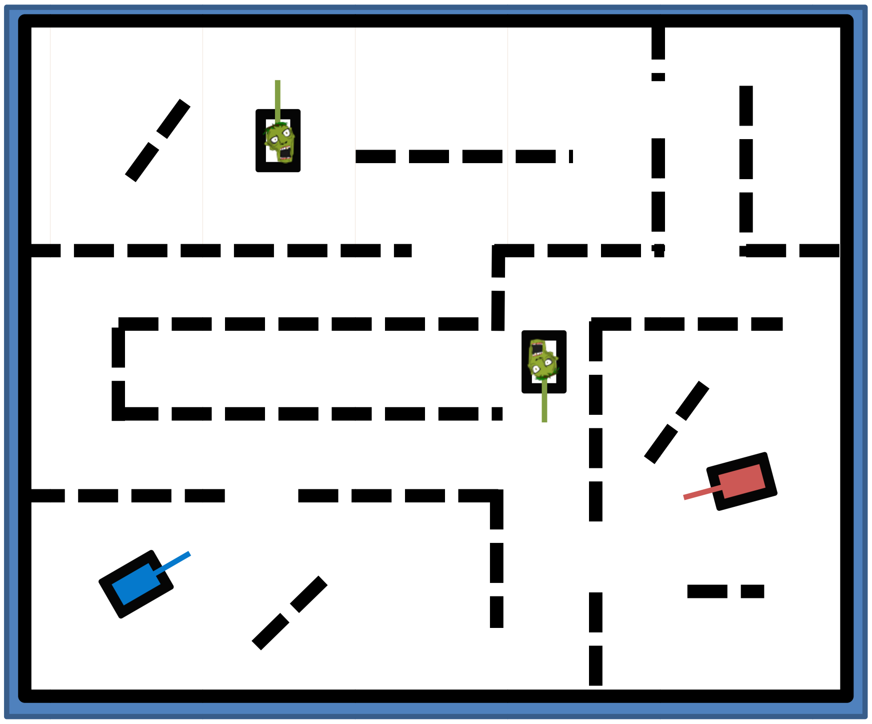
For this question you are to write a C# program using Unity to play a simple networked multiplayer version of Tank Warfare. Your program must comprise of the following:

R1) Level Environment

- Using Unity as your game engine, you will modify your level from Assignment 2 as follows:
  - Scale the geometry of the level to allow tanks to move around.
  - Remove any small rooms off of larger rooms.

- Replace any non-linear objects in the rooms with linear obstacles.
- Represent the walls and any linear obstacles with rows of regular partitions. Removing the partition should make a hole in the wall as wide as the partition.

- Below is an example level showing the minimal requirements.



R2) Networked Multiplayer Tank Warfare Game:

- Starting with the example shown in course lab in Week 8 or by using Unity Networking, create a networked multiplayer 2.5D Tank Warfare game. Each player will control a tank with all the players playing in the same maze (all the connected players should see the same updated level). The objective of the game is to have the tank that you control to shoot all the other tanks. Any tank round that hits a wall or linear obstacle partition will destroy that partition. On the client for the player, if a tank round is fired or hits something and destroys it, play a sound effect. Collisions between tanks are ignored.
- Near the four corners of the maze place four flashing tank rounds representing power ups (such as allowing the tank rounds to bounce off of wall partitions; your choice). If the tank "eats" a power up, then they will temporarily acquire the special ability.
- The movement of the tank is restricted to the four orthogonal directions and can be controlled by the player using the A-S-W-D keys or arrow keys. You can additionally add the ability to rotate the tank on the spot. The tank cannot destroy walls by coming in contact with the walls of the maze. The starting position of the tanks is up to you.
- Implement two to four zombie tanks. The movement of these zombie tanks is up to you but must be deterministic (no random choices) and react to the positions of the tanks of all the players (not just the tank on the client). If a zombie tank gets within a certain range of a client tank, it will start shooting

rounds in the direction of the client tank if it has line of sight. The zombie tanks can't be destroyed.

- The Lab example uses a non-authoritative server to allow for a peer-to-peer connection. Consider the game state information that needs to be updated for the other connected players' clients. You only need to test and demonstrate your game with two players on separate computers. One of the goals of your implementation is to share as little information as possible between clients.

EVALUATION CRITERIA For Question 3 of this assignment

1. Only working programs will get credit. The marker must be able to run your program if it works to evaluate it, so you must give any instructions necessary to get your program running. If your program does not run, we will not debug it.
2. Breakdown:
   - R1: 10% (divided among the requirements listed in item R1 above) [Note: if you do the assignment in 2D using sprites, you will not get full marks]
   - R2: 50% (divided among the requirements listed in item R2 above)
   - Appropriateness of level design and general aesthetics : 10%
   - Source program structure and readability: 5%
   - Write-up: 5%

What to hand in for Assignment

Questions 1-2: (20%) (Theoretical Questions)
- Submit your answers to questions 1 and 2 on the Moodle course webpage (as a MS Word or PDF file, for example, called **TheoryYourIDnumber.doc** or **TheoryYourIDnumber.pdf**). Submit this file as "Theory Assignment 3".

Question 3 (80%) (Practical Question)
Submission Deliverables (Only Electronic submissions accepted) :

1. Submit a well-commented C# program for Unity including data files, if any, along auxiliary files needed to quickly get your program running, and any other instructions for compiling/ building/running your program. The source codes (and brief write-up, explained below) must be submitted **electronically** in a **zip format** with all the required files (example: **YourIDnumber.zip**). Submit this zip file as "Programming Assignment 3". ***Please do not e-mail the submission.***
2. Demonstrate your working program from an **exe file dated on or before April 2 at 23h55** to the lab instructor in the lab during the periods of April 3-5.
3. Include in your zip file a brief write-up about your program, explaining any special features or implementation details that you would like us to consider during the evaluation.