Dept. of Computer Science & Software Eng., Concordia University
COMP 476 --- Winter 2021

Advanced Game Development

Assignment 3 ---- Due Sunday, April 5, 2021

PURPOSE: This assignment will give you the opportunity to learn programming a networked multiplayer game. This assignment contains both **theoretical** questions, which you do not 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: (10%) [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 R R R L L R L R L R L R R L R R L R L R L L L L R L R R

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

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

c) (2%) Using a hierarchical 3-gram predictor, what is the predicted next action for input "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.

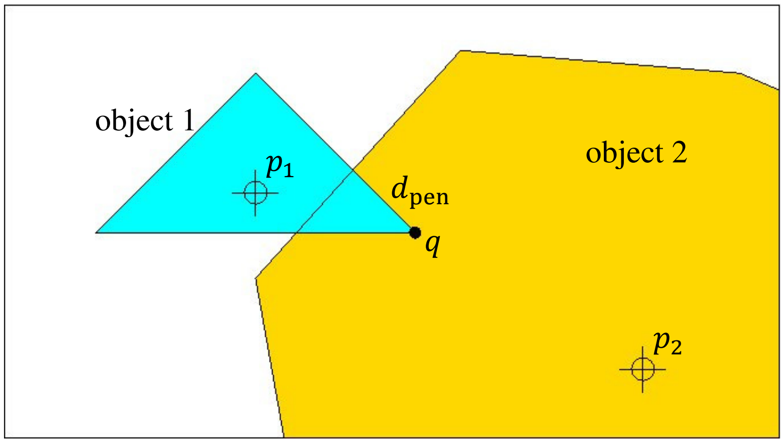
Question #2: (5%) [Theoretical Question]

Consider a sphere with radius 3 and center point (1, 1, 1) colliding with triangular face of an object. Vertices of the triangle are (-3, -3, -7), (3, -3, -1), and (-3, 3, -1). Determine the following:

a) (3%) the contact normal; and

b) (2%) the interpenetration depth.

<u>Question #3:</u> (10%) [Theoretical Question]



Consider the following contact data from the above interpenetration of two objects (object 1 being the side view of a right circular cone):

Centre of mass for object 1, $\boldsymbol{p}_1 = (12, 15.75, 0)$
Centre of mass for object 2, $\boldsymbol{p}_2 = (29, 8, 0)$
Mass for object 1, $m_1 = 2$
Mass for object 2, $m_2 = 51.528$
Contact point, $\boldsymbol{q} = (19, 14, 0)$
Contact normal, $\boldsymbol{d} = (1, -1, 0)$
Penetration depth, $d_{\text{pen}} = 3.8$

Note that the inertia tensor for object 1 (a right circular cone with radius 7 and height 7) is

$$I_1 = \begin{bmatrix} 36.75m_1 & 0 & 0 \\ 0 & 14.7m_1 & 0 \\ 0 & 0 & 36.75m_1 \end{bmatrix}$$

and the inertia tensor for object 2 is

$$I_2 = \begin{bmatrix} 389m_2 & 0 & 0 \\ 0 & 389m_2 & 0 \\ 0 & 0 & 778m_2 \end{bmatrix}$$

Perform the following calculations (show your work) to use nonlinear projection to resolve the interpenetration.

a) (4%) Find the linear and rotation components, $(I_l)_i$ and $(I_a)_i$, of the inertia for both objects.
b) (1%) Compute the total inertia, I_t .
c) (1%) Compute the amount of linear movement $(\Delta_l)_i$ for both objects.
d) (2%) Compute the amount of angular movement $(\Delta_a)_i$ for both objects.
e) (2%) Compute the total rotation , $(\Delta\boldsymbol{\theta})_i \left(\frac{(\Delta_a)_i}{(I_a)_i}\right)$, to be applied to the orientation quaternions for both objects to obtain the required amounts of angular movements.
f) (BONUS: 2%) Assuming the right circular cone has an orientation quaternion of $\theta_1 = [1, 0, 0, 0]$, and noting that this orientation quaternion is altered by the total rotation $[0, (\Delta\theta)_1]$ (computed in e), expressed in quaternion form) by the formula

$$\theta_1 = \theta_1 + \left(\frac{1}{2}\right)\left[0, (\Delta\boldsymbol{\theta})_1 \left(\frac{(\Delta_a)_1}{(I_a)_1}\right)\right]\theta_1$$

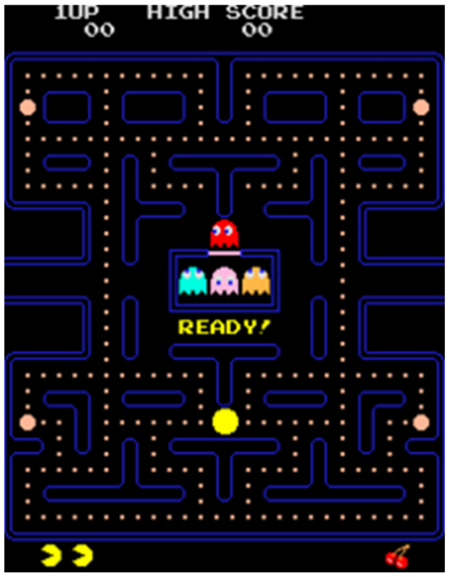
(using quaternion math), what is the change in orientation of the cone in Euler angles? (Show some work…)

Question #4: (75%) [Practical Question]

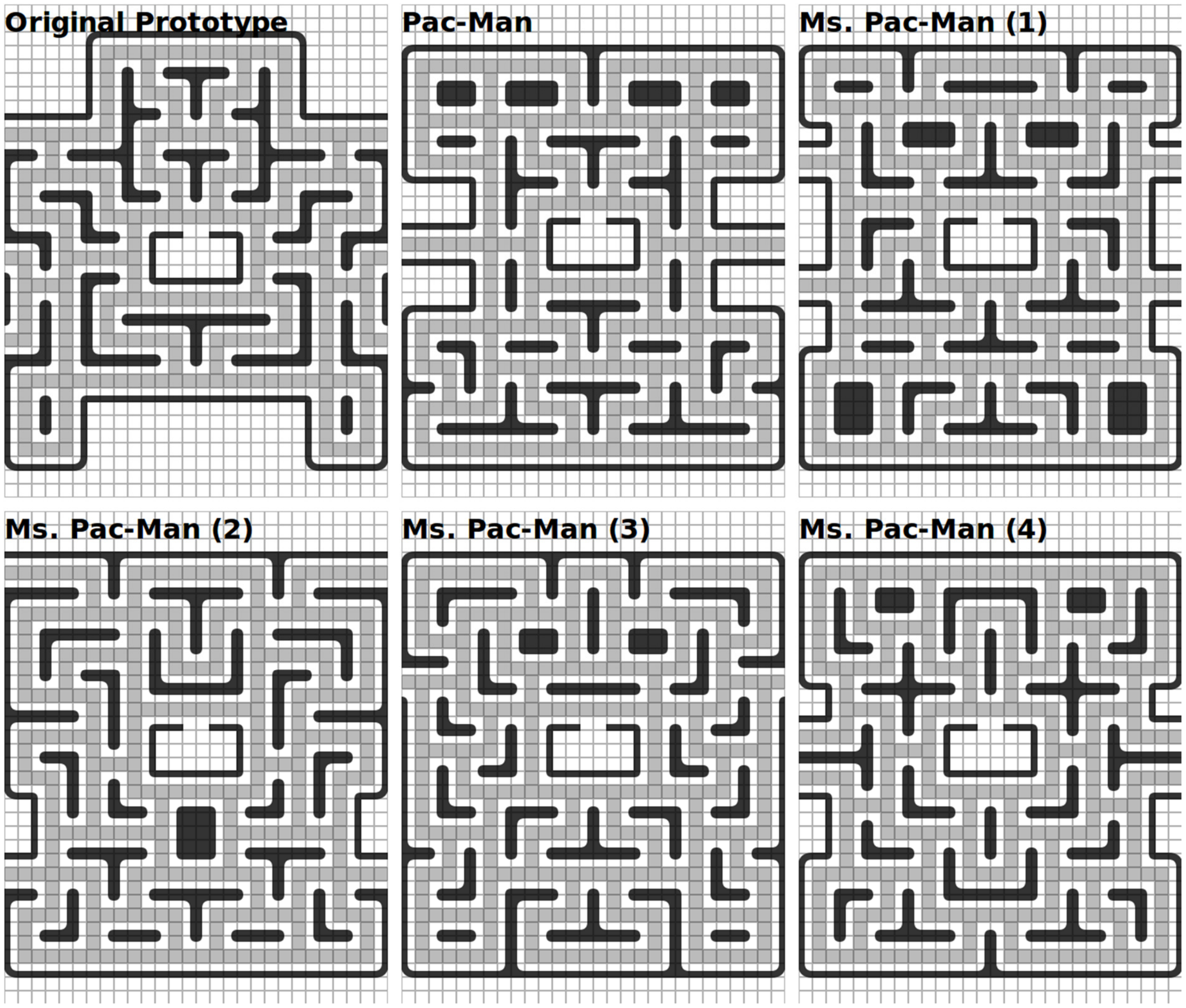
Problem Statement:

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

R1) Level Environment



- Using Unity as your game engine, you will create a Pacman style 2.5D maze level. The level should be based on a 28x31 tiling and the paths are one tile thick with no sharp turns. The exit tunnels should wrap around as in the toroidal space using for Assignment 1. Example maze levels are shown below from the Pac-man and Ms. Pac-man games. Along the paths, at regular intervals, place pac-dots (or pellets) that your player character (PC) will "eat" (see the screenshot to the right; source: en.wikipedia.org/wiki/Pac-Man). Have the pac-dots hover a fixed distance above the "floor" in the 2.5D maze.



[Source: pacman.shaunew.com/play/mapgen/]

R2) Networked Multiplayer Pacman Game:

- Based on the examples shown in lab, create a networked multiplayer 2.5D Pac-man game. Each player will control a “Pac-man” with all the players playing in the same maze (all the connected players should see the same updated level). Note that in 2.5D, the overall shape of the Pac-man is of a hockey puck, not a sphere. The objective of the game is to have the PC that you control eat more pac-dots than the other players by the time all the pac-dots are eaten. A pac-dot is eaten if a collision between your PC and the pac-dot is detected. On the client for the player, as the PC eats a pac-dot, play a sound effect. Collisions between PCs are ignored.
- Near four corners of the maze place four larger, flashing dots representing power pac-dots (also known as power pellets; see figure on previous page). If the PC “eats” a power pac-dot, then their speed will temporarily increase.
- The movement of the PC is restricted the four orthogonal directions of the underlying grid and can be controlled by the player using the A-S-W-D keys or arrow keys. The PC should never come in contact with the walls of the maze. The starting position of the PC is up to you, but the easiest (and fairest) approach is to have all the PCs start from the same position (the original starting position in the original Pacman game; see figure).
- Implement two to four “ghosts” NPC characters (see figure on previous page). The movement of these ghosts is up to you but must be deterministic (no random choices) and react to the positions of the PCs of all the players (not just the PC on the client). If a ghost touches a PC, then the PC goes back to its starting position for that client. The ghosts can’t be eaten even if after eating a power pac-dot.
- 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 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 4 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: 20% (equally 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: 35% (equally 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 1-3: (25%) (Theoretical Questions)
- Submit your answers to questions 1 and 2 electronically through Moodle (as a PDF file, for example, named, **A3TheoryYourID.pdf)**: Submit this file using Moodle under "Theory Assignment 3" by Monday April 5, 2021 at 23h55.

Question 4 (75%) (Practical Question)

Submission Deliverables (Only Electronic submissions accepted):

1. Submit a well-commented C#/Unity source program 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: **A3YourID.zip**): on Moodle under the "Programming Assignment 3" link. ***Please do not e-mail the submission.***
2. Demonstrate your working program from an **exe file dated on/before April 5 at 23h55** to lab instructors during your assigned lab period of April 6 to April 9.
3. A brief write-up about your program, explaining any special features and/or implementation details that you would like us to consider during evaluation.