Ethan Alward ID: 3695438

March 27th, 2025

Concept Assignment #3

CS2053 Introduction to Game Development

1. Complete the pseudocode below to detect if two spheres have collided.

```
struct Sphere3D {
	float radius;
	//coordinates of the center point
	float x, y, z;
}
bool SpheresCollide(Sphere3D s1, Sphere3D s2) {
	//recall that the Pythagorean theorem in 3D is sqrt(a*a + b*b + c*c)

	//if their two center points are closer together than the length of both radiuses combined, they are touching

	var distanceBetweenCenters =
sqrt( pow(abs( s1.x - s2.x), 2)  +  pow(abs( s1.y - s2.y), 2)  +  pow(abs( s1.z - s2.z), 2) )

	//returns true if spheres are colliding, false otherwise

	if distanceBetweenCenters <= (s1.radius + s2.radius):
		return true
	else:
		return false
}
```

2. In games, collisions between characters are usually not very precise, as oftentimes the arm or leg of one character will penetrate into another character. If precise collision detection is employed, what problems or issues might arise? What advantage does modeling character-character collisions with spheres or cylinders/capsules offer?

If precise collision detection is employed a character finger hitting a door frame could prevent the character from walking through the door. This would make the gameplay worse for the majority of users who don't care about the importance of perfect character movement. Modelling character collisions with spherical shapes prevents corners and edges from poking out and catching/colliding.

3. Before a ball hits the wall it's speed is 150 units/second, after it hits a wall the speed becomes 200 units/second. Calculate the Coefficient of Restitution. What type of collision would this be when considering this coefficient? In what game scenario might this make sense?

$$Coefficient\ of\ Restitution\ =\ \frac{velocity\ of\ separation\ after\ collision}{velocity\ of\ approach\ before\ collision} \qquad \frac{200\ units/second}{150\ units/second} = 1.\overline{33}$$

When the coefficient of restitution is greater than 1 a superelastic collision has occurred. I imagine a baseball being thrown from a pitcher and the batter hitting back even harder than it came in. Bouncing on a trampoline could be another similar example of this.

4. For the picking problem (using a mouse to select an object in a 3D world), after applying the unprojection matrix (by inverting matrix multiplication of camera matrix and projection matrix), we can compute the picking point A0 on the near plane (Z=0) and picking point A1 on the far plane (Z=1) in the 3D camera view. What else needs to be done to select a 3D object based on this approach?

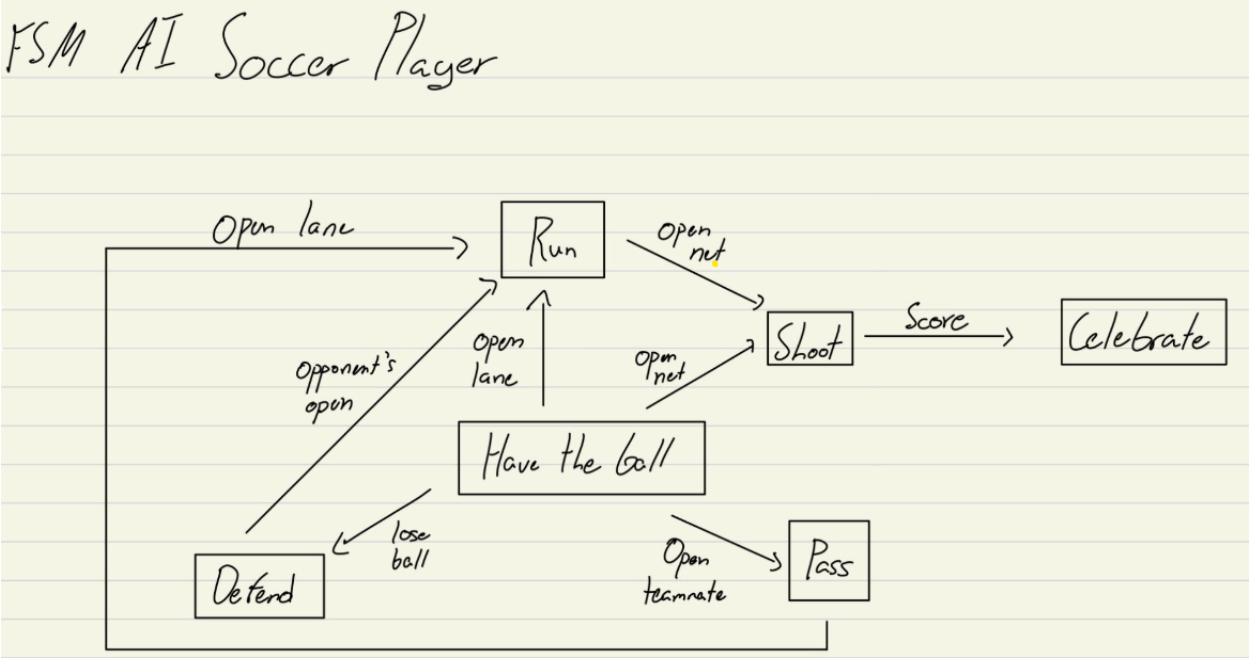
When ray casting you get the 3D world's space state, then you get the viewport's mouse position, then you get the ray's origin, and then the ray's end. Using the ray's origin and end you check if it's intersected with the space_state. To check what you've clicked on you get the intersection of your raycast and a node.

I've used the following code for checking if a user clicked on a square in my chess game:

```
# ray casting
if Input.is_action_just_pressed("click"):
	var space_state = get_world_3d().direct_space_state
	var mouse_position = get_viewport().get_mouse_position()
	rayOrigin = $Camera3D.project_ray_origin(mouse_position)
	rayEnd = rayOrigin + $Camera3D.project_ray_normal(mouse_position) *
1000
	var ray_query = PhysicsRayQueryParameters3D.create(rayOrigin,
rayEnd)
	var intersection = space_state.intersect_ray(ray_query)
	if intersection:
		squareClicked = intersection["collider"].get_parent().get_parent()

		if squareClicked.is_in_group("square"):
		//yada yada find out which square it is
```

5. Design a FSM (Finite State Machine) for behavior of an AI soccer player. The FSM should have 4 or more states and state transitions between them based on some events. Draw a diagram similar to the one below to represent the potential states of the player. Provide details about the transition on the arcs between states.



After shooting the ball if a goal event occurs the soccer player will then celebrate. The other states are soccer gameplay related. If I have the ball and I'm open near the net I'm gonna shoot, if my teammate is open I'm gonna pass. If I lose the ball I will start playing defense.

6. Why is A* typically preferred over Best-First and Dijkstra algorithms?

A* is preferred because it's efficient. It uses a heuristic function h(x) unlike Dijkstra, and it calculates the cost of going from the start node to the current node while also having a path-ost component g(x).

7. Write pseudocode for how you might provide localization in 4 different languages, including what the data structures might look like. Show how the code and data might be implemented for a button that says "Start", keeping in mind that your data structures would need to be scalable for hundreds or thousands of text elements.

The first trick that comes to mind is setting the custom minimum size's x component to fit the longest a word is in the 4 languages. So if COMMENCER is the longest version of START I would set the button's min size to fit COMMENCER. In the notes it says we should use a dictionary to hold the strings and have the key be the language.

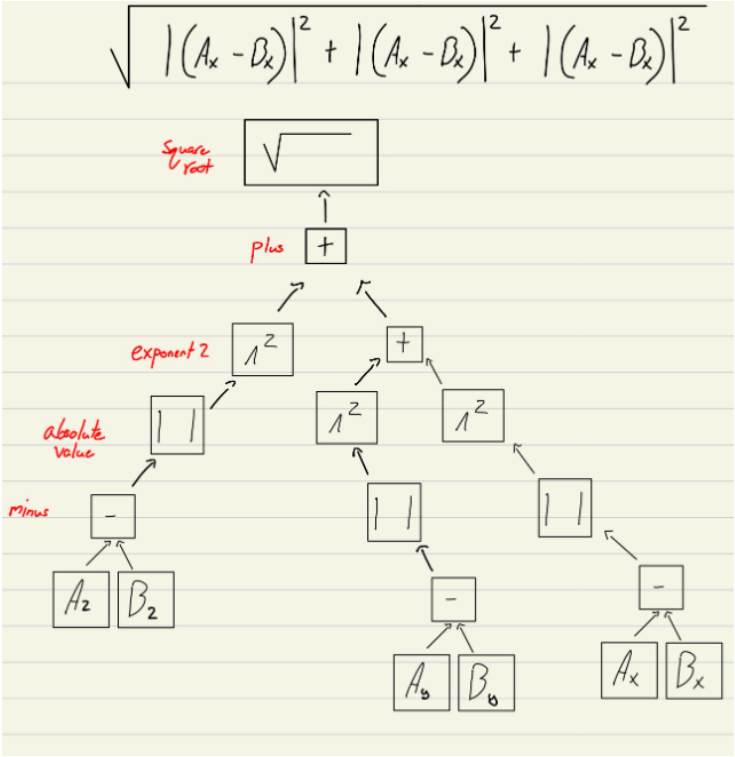
var startButtonStrings = { "English" : "START", "French" : "COMMENCER", "Spanish" : "COMENZAR", "Chinese" : "开始" }

var allStrings = { "start" : startButtonStrings, "end" : endButtonStrings …etc}

var curLang = “French”

//in Start button script
//set the button text to the selected language
StartButtonText.text = allStrings.get(“start”).get(curLang)

8. Draw an abstract syntax tree (AST) for calculating distance between 2 different 3D points, A and B. You can consider their x components Ax and Bx, etc. I'm looking for a mathematical AST, not code.



9. You are playing an MMORPG that has 3 dedicated servers in different parts of the world. Describe the process by which your computer would choose which server to connect to, including how data is sent and what data is sent.

My understanding from the games I've played, the client will ping test all the servers and choose to connect to the one with the lowest ping. The ping is just the time it takes to send and receive a tiny amount of data from the server. MMOs like World of Warcraft use a TCP connection which is established using a three way handshake where the client sends a SYN, server responds with a SYN ACK, and the client responds again with an ACK, then the connection is considered established.