CW1 Tutorials

Tutorial 1: Edge Detection AI

This tutorial is for AI simple movement, rotation and more importantly giving the AI an ability to detect if it will fall. The edge detection is rather simple, however, if you want to make your own pathing code this may be useful as starting code.

Specific program info:

* AI Movement
* Edge detection when reaching a side of the object
* Changing which object to check for fall when landing on a new object

Firstly, we must make the simple AI object. For this we take a cube, add an Rigidbody (RB), and make sure it has the Box Collider. As well as this, create a large flat plane using another box which will be the floor. This large plane will be what the AI uses to move on. This object also needs a Box Collider, but not an RB as it does not contain physics. For this project there is no need to use the RB physics as we are not having any physics as a mechanic. I have attached an RB to the AI object only so that we can test that it will not fall, if it had no RB then there would not be any gravity or mass to make it fall.

After setting up the scene like this, we need to make the simple AI movement. First we define the “speeds” for movement, make 2 public (this can be private, public only so you can edit in inspector) float variables: 1 for forward velocity and 1 for turn rate.

Now to make the simple movement. Create 2 private void functions, RotateObject and MoveInDirection. Both functions should inherit a Vector3 called Direction.

We will code the movement function first. For this project I am using transform movement as it is very precise, although if you wanted you could use RB.AddForce(). Specifically, I used transform.Translate in this function. Translate here means to move in a direction, you -could- use transform.position = new Vector3(…) however this will be highly complicated due to the rotation of the AI object. Inside MoveInDirection, write: transform.Translate(Direction \* Vel \* Time.deltaTime);

transform = object positioning variables

Direction = Vector3 direction, ie: transform.forward

Vel = Speed

Time.deltaTime = makes sure the movement is not affected by frame rate

Now to code the RotateObject. This is much more simple, I used the simple built in rotation code for Unity, transform.Rotate(). We only need to change rotation in 1 axis, the y axis for this tutorial as we are only doing edge detection on a flat surface. Simply write the code: transform.Rotate(new Vector3(0, Direction[1], 0));

Detection of if the AI is touching the floor is rather simple, use the OnCollisionEnter function and check that the collided object has the “floor” tag, this marks it as a valid pathing object. After colliding with a floor, the code calls the MoveLogicInit function. This immediately calculates and stores the boundaries of the object with 4 variables to determine where not to move the AI past.

For the next part of the code it is best to look at the in-code comments to explain what it does.