DH2323 Project Report

Markus Wessén, mwesse@kth.se, 010209-4653

Introduction

First of all, I did the project on my own. My project is expanding lab 3 of the animation track in unity. The idea was to transform the tank into a walkable robot by rotating and moving the parts of the tank. The goal was to have the wheels and the tracks be the legs and the body of the tank be the body as well as the turret as the head. After rotating and moving all parts into place, you can walk around with the robot where the legs swing back and forth. Also the turret will then not be locked in the y-direction, allowing for shooting downwards towards the tanks.

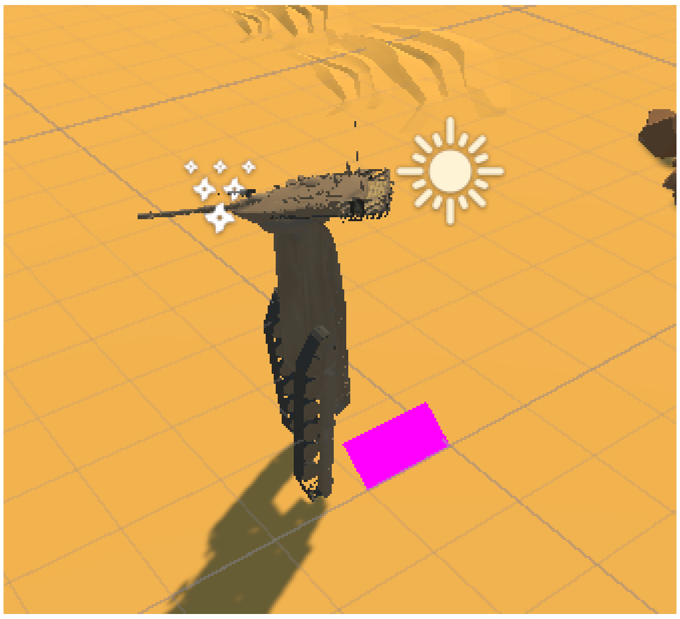
Implementation

At first glance, the task seemed easy at hand. Simply rotate each part around an axis and translate the body and turret appropriately upwards. However, the given tank in the scene is made up of several parts where each part has its own local coordinate system. This means their axes point in different directions ultimately meaning it is not possible to call the same rotate function on all parts to rotate the tank since they rotate around their own local axes. My first idea to solve this problem was to instead use the world axes and rotate the entire tank around a world axis. This works in theory if the tank has not moved from its starting position. However, the world axes are constant which means: if the tank moves from its start position, the relation between the tank and the world axis to rotate around will change making it rotate in a different way. Thus, the solution used was finding the forward direction of the tank using the turret. Using that direction it was possible to find the direction pointing right of the tank, which was the axis used to rotate the tank around. All of this also meant it was not possible to use quaternion based rotation around each local origin. The solution to this was I found a function in the unity documentation called RotateAround [1]. This function allows you to choose a point and an axis from that point to rotate around. Thus I could choose a point and direction in the middle of the tank and rotate all parts (except the turret) around that axis to make it stand up.

After rotating everything, since the tank was stationed at the ground, the parts rotated partially into the ground. So before all rotations I had to move the entire tank upwards a certain amount to prevent the parts from rotating into the ground. After the rotations were done and the body's front was pointing in the global y-direction (upwards). Moving the body and the tank upwards was easy since the direction to translate was just the global y-axis. To make the translation smooth I used linear interpolation [2] between the current starting point of the parts and a set ending point as well as using "yield return null" to ensure each iteration happened once per frame instead of the entire procedure happening in one frame. This ensured the parts only moved a certain amount each iteration instead of moving the entire way instantly. This was also how the initial movement upwards was made before all the rotations happened.

Final result

https://drive.google.com/file/d/1_e8OWIV822ACsRMCcvrLO8qy93UF26i5/view?usp=sharing



What I learned

I learned a lot about unity and how unity works for instance. But I also got a better understanding of the logic of the relation between different axes and local coordinate systems. Having to figure out that just rotating a part locally would not work and that the relation between the parts meant you had to resort to global rotating schemes. I also learned that

User studies

User studies could provide valuable insights into how users perceive the transformation from a tank to a walking robot. For example, a perceptual study might reveal whether users find the transformation smooth or choppy. Based on such feedback, improvements could be made to make the leg motions slower and smoother to enhance the sense of realism and not just parts moving.

Additionally, a user study might show that users expect knees to bend when the robot walks as well as the “feet” touching the ground when walking. This could justify splitting the parts at the legs into two and adding rotations at the mid-points to simulate knee motion. This could also mean adding a gravity aspect to make the robot lean forward and move down making the legs touch the ground when walking instead of floating. For realism sakes, it would not be unrealistic to get feedback about the fact the robot is missing arms. Considering the tank does not have extra parts initially that could be used for parts it makes sense the robot does not have arms. But to make it more realistic it could be considered to add temporary parts when transforming into a robot that could be used as arms. Or potentially if it exists, use parts from the legs or body as arms instead of using them for the legs or body.

A curious user could possibly ask about the implementation of the rotation, especially about the translation of the parts. In that case it is possible to figure out that the translation was hard-coded and thus not directly translatable to different types of tanks. In that case it could be a good idea to make the translation values dependent on the size of the parts and calculated instead of hard-coded values obtained from testing. This allows for the implementation to possibly be shared and used in other cases without large changes in the code. This could be other types of tanks with other parts for instance.

References

[1]https://docs.unity3d.com/6000.1/Documentation/ScriptReference/Transform.RotateAround.html
[2] https://docs.unity3d.com/ScriptReference/Vector3.Lerp.html