DH2323
Computer Graphics and Interaction
Lab 3: Collision Detection and Level of Details

Viktor Meyer viktorme@kth.se

June 15, 2020

1 Introduction to the Unity physics system

1.1 Colliders

The Unity physics system supports colliders that allow the runtime to detect collisions between game objects. Colliders mainly come in the form of primitive types such as boxes, spheres or capsules. It is worth noting that it is also possible to use mesh colliders that can use an underlying 3d models mesh as reference.

Static colliders are as the name implies static and are not affected by rigidbody forces. This makes static colliders useful for scene objects that never moves such as level geometry. When a collider is attached to a rigidbody, the forces from the collisions that the collider experiences are applied directly to the underlying rigidbody. As a result from this, a game object with a rigidbody and collider will move when acted upon by an external object.

1.2 Triggers

Triggers are a setting on collider componenents that control their behaviour. If a collider is set to trigger mode the physics system will ignore that collider. However, it is still possible to listen other events that are related to the collider. Examples of this are the OnTriggerEnter() and OnTriggerExit() callbacks that fire when a collider intersects another. This is useful when designing functionality that requires knowing whether a game object is contained within another.

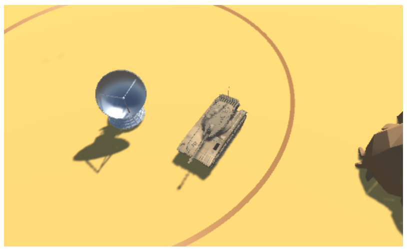
1.3 Raycast

Raycast involves testing which objects a ray intersects in the scene. A ray is defined by an origin and a direction. Raycasts return true if they hit a collider and false otherwise. Sometimes it is required to know more information about what collider the a ray hits, this can be done through the RaycastHit object.

2 Tank detection and firing

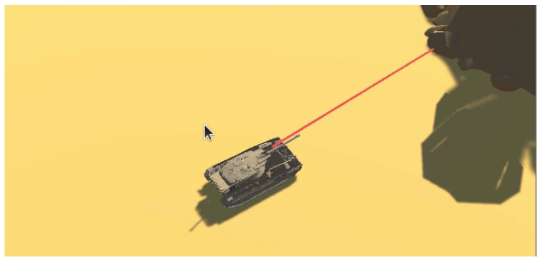
2.1 Radar detection

Radar detection requires the functionality of detecting whether the tank game object is contained within the radar range. This makes a good use case for the OnTriggerEnter() and OnTriggerExit() callbacks. Once a game object enters the radar collider's trigger, the radar first checks whether the triggering object has the "Player" tag. If the player tag is encountered, the radar material is switched to red and alarm is triggered. Once an object exits the radar collider's trigger, the radar does another check to see if the exiting object has the "Player" tag. If the player tag is encountered, the radar material is switched back and alarm is cancelled.



2.2 Tank secondary gun

Raycasting is instantaneous which is why it can be used to simulate infinitely fast bullets from weaponry such as machine guns. Shooting the secondary gun of the tank can be implemented by listening for mouse input and using a timer to control firing rate. For each shot a raycast operation is performed to check whether the shot hits anything. A layer mask can be used to control what the tank can hit with its secondary gun. In this case a layer mask "Shootable" defines what can be hit. If the ray hits anything that is shootable, a line renderer can be used to illustrate the bullet trajectory. If the ray does not hit anything that matches the layer mask, it is usually good practice to still show the line renderer but with its endpoint set to the rays maximum length/endpoint.



2.3 Tank main gun

When it comes to firing the main gun raycasting is not a good choice since the heavier projectiles tend to have air time. Work from the previous lab can be used where we relied on rigidbodies and the physics system for simulation. If mouse input is detected a projectile is instantiated and force is applied to its rigidbody which makes it move. The projectile itself listens for collisions and upon collision checks whether it has collided with an enemy. If an enemy is detected it is destroyed and explosion effects are spawned.

3 Level of detail

Level of detail can be used to reduce the rendering complexity of a scene. The strategy is to render objects that are further away from the camera with less detail than those that are close. Usually this is done by having artists build multiple meshes at different resolutions for each model.

In Unity it is very simple to use level of detail since there is pre-built functionality for it. LOD Groups can be configured to use a high resolution models for the enemy tanks when they are close to the camera and low resolution models when they are far away.

The level of detail behaviour can be observed in the image below where the range of a particular level of detail changes at the red line. It is possible to see that the tanks closer are of different model than the ones further away. The transition is obvious, however, this is simply because the high and low resolution models are not aligned/ideal.