EIE3360 Integrated Project
Laboratory 1A – Game Setup & Alarm Logic

1. Objective

In this laboratory exercise, you will use Unity to build a third-person stealth game for Android. You need to develop the following environment setup in this lab,

1. **Game Setup and Lighting**
2. **Alarm Light**
3. **Tag Management**
4. **Game Controller**
5. **CCTV Cameras**
6. **Laser Grids**

2. Preparation

- A PC with Microsoft Windows
- Unity Hub
- Unity 2022.3.16
- Android SDK
- Microsoft Visual Studio
- Basic knowledge in C# programming, object-oriented programming and 3D graphics

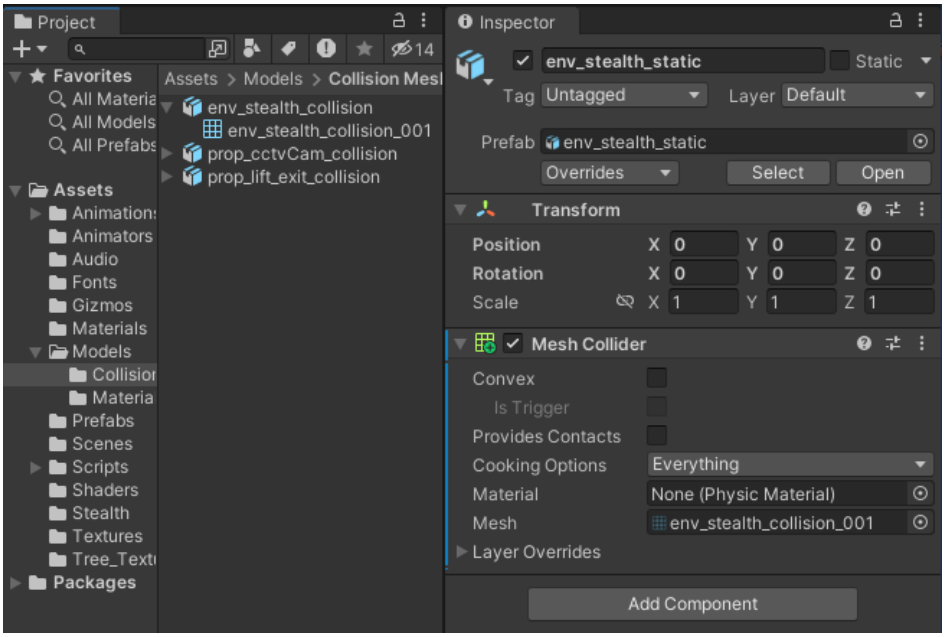
3. Developing a Unity Android Game

This laboratory exercise uses Unity to create a game called 'Stealth' for Android. In the game, the player sneaks around to avoid triggering the alarms and tries to escape from a maze where enemies are patrolling.

3.1 Game Setup and Lighting

1. **Create a new Unity Project**
   a. Create a Project named "**EIE3360Lab1A"**.
   b. Download the "Stealth" package from Blackboard and import the package to the Assets folder.
   c. Save the Sample Scene as "Stealth_Lab1A" in the Assets folder.
   d. Delete the Directional Light in Stealth_Lab1A
   e. Window -> Rendering -> Lighting -> Environment tab -> Skybox Material -> None.
   f. Click File → Build Settings, click "Add Open Scenes" to add the "Stealth_Lab1A" scenes to "Scenes In Build", and make sure the index is 0.
   g. Save the scene and the project.
2. **Create a "Game Scene."**
   a. Go to the "Models" folder in the Project window and drag and drop "env_stealth_static" into the Hierarchy.
   b. Ensure the position is X = Y = Z = 0.

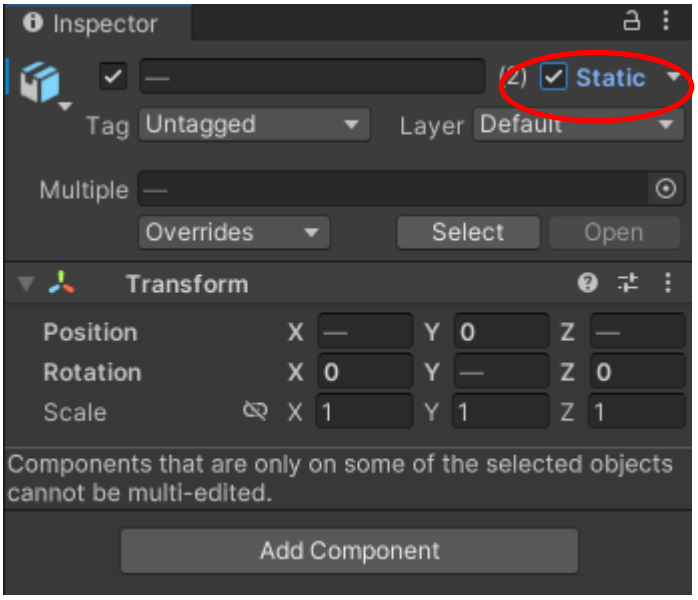
c. Drag and drop “prop_battleBus” into the Hierarchy.
d. Set the position to X = -11, Y = 0, Z = 17.5.
e. Rotate it to Y = 270.
f. Focus on the bus.
g. Select “env_stealth_static”, and add a “Mesh Collider” in the Inspector.
h. In the “Models\Collision Meshes” folder, expand “env_stealth_collision” and drag the “env_stealth_collision_001” to the “Mesh” properties of the “Mesh Collider”.



3. A mixture of baked dynamic light

Making the environment “static” allows game objects to be light mapped.

a. Select both “env_stealth_static” and “prop_battleBus”, and check the “Static” check box in the top-right of the Inspector.



b. Click “Yes, change children”.
c. Expand “env_stealth_static”, select every child object except “extents”, also select “prop_battleBus” and add a “Layer” and name it as “PlayArea” in the “User Layer 8”.
d. Click “Yes, change children”.

e. Select the “extents” game object in the Hierarchy and add a “Layer”, and name it “Extents”. Click “Yes, change children” again.

f. Save the scene and the project.

4. Set the camera

a. Select the “Main Camera” and rename it to “Camera_Main”.

b. Change the “Clear Flags” to “Solid Color” in the Inspector.

c. Change the “Rendering Path” to “Deferred”.

d. Set “Clipping Plan” Far to “100”

e. Click the color box of the “Background” and set the color to Black (R=G=B=0).

5. Add the lights

a. Go to the Prefabs folder in the Project windows, drag and drop the “lights_baked” game object to the Hierarchy and reset the position to X = Y = Z = 0.

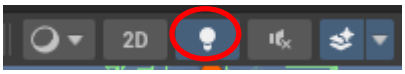
b. Create a “Point Light” in the Hierarchy and rename it to “light_playArea_point”. Set the position to X = -3, Y= 1, Z = 10.

c. Adjust the light color to R = 255, G = 245, B = 150, A= 255 and reduce the “Range” to 8.

d. Change the “mode” from “Realtime” to “Baked”.

Note: This can avoid rendering in real time to affect performance

e. Turn on and off the lighting by clicking the “Light bulb” icon at the top of the Scene windows if needed.



Note: You can turn off the lighting by clicking the Light bulb icon again

f. Duplicate the “point” light two times, and move one to the center of the room and the other one to the end of the room.

g. Drag the three “point” lights to the “lights_baked” game object to keep them organized in the Hierarchy.

h. Save the scene and the project.

6. Change the render setting

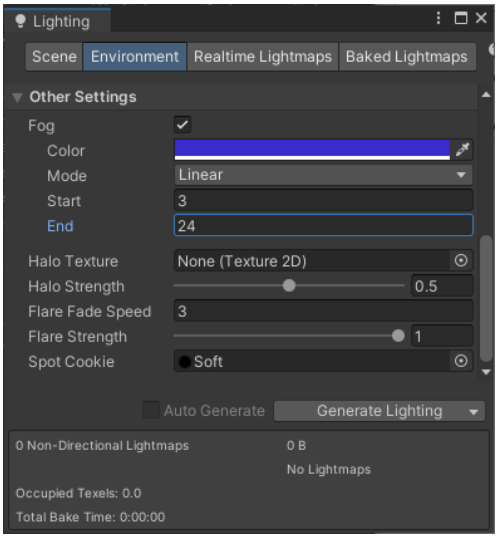
a. Click Windows → Rendering → Lighting Setting and go to the “Environment” tab.

b. Go to Other Settings → check the “Fog” to enable it.

c. Set the “Fog” Color to R = 57, G = 43, B = 204, A = 255.

d. Set the Mode to “Linear”.

e. Set the Linear Fog “Start” to 3, and “End” to 24.

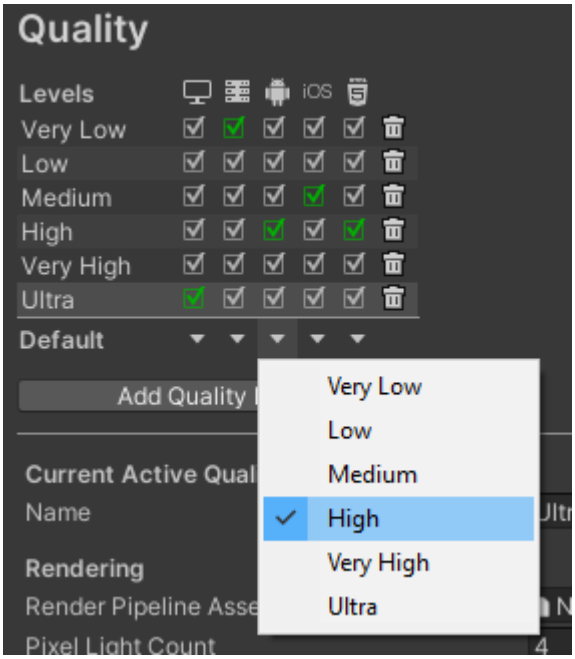


f. You may turn off the fog in the Scene to have a better view during development.

g. 

7. Change the quality setting and bake light

1. Click Edit → Project Setting → Quality.
2. Click on the arrow under the "Default" of the Android platform and make it "High".



8. Create the directional light and alarm light

a. Create a "Directional Light" from the Hierarchy and rename it to "light_main_directional".
b. Add a tag and name it "MainLight" in the Inspector and assign the "MainLight" tag to "light_main_directional".
c. Change the light color to R = 33, G = 45, B = 48.
d. Set the "Shadow Type" to "Soft Shadows".
e. Click the "Culling Mask" and de-select "Extents".
f. Set the Rotation to X = 45, Y = 305, Z = 0.
g. Duplicate the "light_main_directional" and rename it to "light_alarm_directional".
h. Add a tag, name it to "AlarmLight" in the Inspector, and assign the "AlarmLight" tag to light_alarm_directional.
i. Set the color to R = 70, G = 0, B = 0.
j. Set the "intensity" to 0.
k. Click the "Culling Mask" and select 'Everything'.
l. Save the scene and the project.

3.2 Alarm Light

1. **Control the alarm light; when the player triggers the alarm, the light pulse will be activated**
    a. Go to Scripts/AlarmSystems in the Project window.
    b. Drag the "AlarmLight" script to the "light_alarm_directional" object.
2. **Modify the megaphone towers:**
    a. Expand "env_stealth_static" → select the "props" game object and expand it in the Hierarchy.
    b. Select "prop_megaphone_001 to 006" and add a Tag and name it "Siren" in the Inspector and assign the "Siren" tag to "prop_megaphone_001 to 006".
    c. Click Add Component → Audio Source.
    d. In the "Audio" folder in the project window, drag and drop "alarm_triggered" to the "Audio Clip" properties.
    e. Uncheck "Play on Awake".
    f. Check Loop.
    g. Expand "3D Sound Settings" and set the "Min Distance" to 5.
    h. Save the scene and the project.

3.3 Tag Management

1. **The tag in Unity is for identifying references to game objects. We will apply a script to the variables for convenience.**
    a. The following script contents are for your reference only.

```
using UnityEngine;
using System.Collections;

public class Tags : MonoBehaviour
{
    // A list of tag strings.
    public const string player = "Player";
    public const string alarmLight = "AlarmLight";
    public const string siren = "Siren";
    public const string gameController = "GameController";
    public const string mainLight = "MainLight";
    public const string enemy = "Enemy";
}
```

3.4 Game Controller

A game controller handles information that must be widely available for controlling objects in the game. Our game controller will be taking a variety of tasks, and the most important piece of information that's stored is the last know location of the player.

1. **To create a game controller:**
    a. Create an empty GameObject and rename it to "gameController". Set the tag to "GameController".
    b. Click Add Component → Audio Source
    c. In the "Audio" folder in the Project window, drag and drop "music_normal" to the "Audio Clip".
    d. Check the "Play On Awake" and the "Loop".
    e. Set the Volume to 0.8.
    f. Create another empty GameObject called "secondaryMusic".
    g. Make it a child of the "gameController" in the Hierarchy.
    h. Click Add Component → Audio Source to add an audio "music_panic" in "Audio Clip".

i. Check “Play On Awake” and the “Loop”.

j. Set the Volume to 0.

a. TWO scripts for the game controllers, one for handling where the player was last sighted and what operation should perform when the player is sighted. Other scripts created in the next session will reference it to know the player’s last sighting.

b. Select “gameController” in the Hierarchy, and drag the “LastPlayerSighting” from Scripts/GameController in the Project window.

c. Play the game.

d. Set the position of the “Last Player Sighting (Script)” to other than ONE thousand.

- You can listen to the panic music begins to play.

e. **Stop** playing the game.

f. Drag the “gameController” to the Prefabs folder in the Project window.

g. Save the scene and the project.

3.6 CCTV Cameras

1. Add a CCTV system that triggers the alarm in the game and the CCTV cameras try to spot the player

a. In the “Models” folder in the Project window, drag and drop the “prop_cctvCam” game object to the Hierarchy.

b. Set the Position to X = -8, Y = 3, Z = 16.1.

c. Set the Rotation to Y = 180.

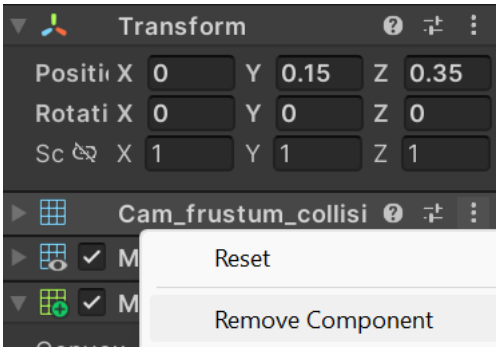
d. Double click “prop_cctvCam” to focus and expand the “prop_cctvCam”. Then, select “prop_cctvCam_body”.

e. In the “Models/Collision Meshes” folder in the Project window, drag and drop “prop_cctvCam_collision” onto “prop_cctvCam_body”.

f. Set the Position to X = 0, Y = 0.15, Z = 0.35.

g. Click Add Component → Mesh Collider and check the box “Convex” and “Is Trigger”.

h. Remove the “Cam_frustum_collision (Mesh Filter)”.



i. Remove "Mesh Renderer" too.
j. Select "prop_cctvCam_body", set the Rotation to X = 60.
k. Select "prop_cctvCam_collision", click Add Component → Light.
l. Change the Type to "Spot".
m. Set the "Range" to 9.5, set the "Spot Angle" to 90, set the "Color" to Red, set the "Intensity" to 8 and set the "mode" to "Baked".
n. Go to "Scripts/AlarmSystems" and drag the "CCTVPlayerDetection" script to "prop_cctvCam_collision" to report the player's location when the CCTV camera spots him.
o. Rename the original "prop_cctvCam" from Prefabs to "prop_cctvCam_bk". Select "prop_cctvCam" in the hierarchy and drop it into the Prefabs folder. Rename the new Prefab to "prop_cctv_Cam_lab1a".
p. Duplicate the CCTV cam TWICE and set the second CCTV camera to the position at X = -21, Y= 2.2, Z = 2, and rename it to "prop_cctvCam_002".
q. Select the third CCTV camera, change the position at X = -23, Y = 1.8, Z = 24, and rename it "prop_cctvCam_003".
r. Double-click "prop_cctvCam_003" to check the location.
s. Expand "prop_cctvCam_003/prop_cctvCam_joint/" and select "prop_cctvCam_body", set the Rotation to X = 30.
t. Select "prop_cctvCam_collision", change the "Scale" to X = 1, Y = 1, Z = 1.8.
u. Rename "prop_cctvCam" to "prop_cctvCam_001".
v. Focus on "**prop_cctvCam_001**".
w. Expand and select "prop_cctvCam_collision" and change the "intensity" to **6**.
x. Save the scene and the project.

Checkpoint 1: Demo to the instructor or teaching assistant

3.7 Laser Grids

1. Create a laser grid to trigger the alarm when the player goes across it

a. In the "Models" folder in the Project window, drag and drop "fx_laserFence_lasers" to the Hierarchy.
b. Rename it to "fx_laserFence_lasers_001", set the Position to X = -8, Y = 1.21, Z = 5.62, set the Rotation to X = 0, Y = 90, Z = 0 and set the Scale to X = 1, Y = 1, Z = 3.6.
c. Double-click the name to focus on it.
d. Click Add Component → Box Collider and check "Is Trigger".
e. Click Add Component → Audio Source.
f. In the "Audio" folder in the Project window, drag and drop "laser_hum" to the "Audio Clip" properties.
g. Set the "Max Distance" to 1.8.
h. Check the Loop.
i. Click Add Component → Light.
j. Change the "Color" to R = 255, G = 40, B = 0.

k. Change the "Range" to 5 and "intensity" to 0.6.
l. Set the "mode" to "Baked".
m. Go to "Scripts/AlarmSystems and drag the "LaserBlinking" script to the "fx_laserFence_lasers_001" object to make the laser "blinks on and off".
n. Drag another script, "LaserPlayerDetection" to manage player detection to the "fx_laserFence_lasers_001" object.
o. Rename the original "fx_laserFence_lasers_001" to "fx_laserFence_lasers_001_bk" in the Prefabs folder and drag and drop the "fx_laserFence_laser_001" in the hierarchy to the Prefabs folder to save it as a prefab. Rename the new Prefab to "fx_laserFence_laser_001_Lab1a"
p. Duplicate it FIVE times, rename them, and set the positions and rotations below.

	Position	**Rotation**
fx_laserFence_laser_002	X = -8, Y = 1.21, Z = 9.23	
fx_laserFence_laser_003	X = -17.93, Y = 1.21, Z = 24.08	
fx_laserFence_laser_004	X = -23.92, Y = 1.21, Z = 26.1	Y = 0
fx_laserFence_laser_005	X = -8.95, Y = 1.21, Z = 25.99	
fx_laserFence_laser_006	X = -8.95, Y = 1.21, Z = 29.96	

q. Set the "Scale" of "fx_laserFence_laser_005" and "fx_laserFence_laser_006" to X = 1, Y = 1, Z = 5.6
r. Set the "On Time" and "Off Time" variables in the "Laser Blinking (Script)" to laser fences 5 and 6 as below.

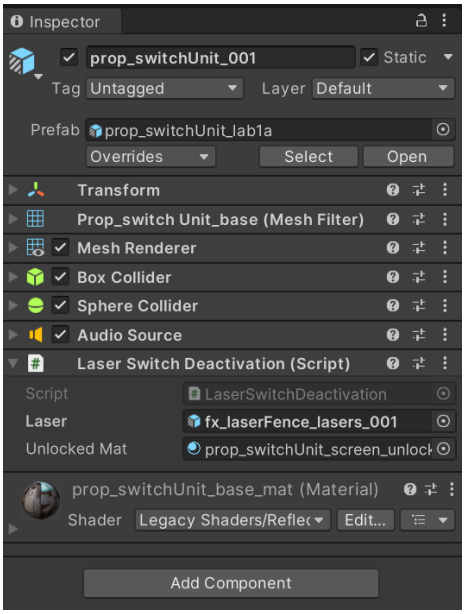
	On-Time	**Off-Time**
fx_laserFence_laser_005	1.5	1.5
fx_laserFence_laser_006	1.5	1.55

2. Create some switches for the laser grids.

a. In the "Models" folder, drag and drop "prop_switchUnit" to the Hierarchy.
b. Set the Position to X = -11.5, Y = 0, Z = 11.3 and set the Rotation to X = 0, Y = 180, Z = 0.
c. Double-click the name to focus on it.
d. Check "Static" at the top right corner, and click "Yes, change children".
e. Click Add Component → Box Collider. Set the "Center" to X = 0.3, Y = 0.8, Z = -0.15 and the Size to X = 1.3, Y = 1.6, Z = 0.8, check the "Is Trigger".
f. Click Add Component → Sphere Collider.
g. Set the Center to X = 0, Y = 1, Z = 1 and the Radius to 1.5; check the "Is Trigger".
h. Click Add Component → Audio Source and uncheck "Play On Awake".
i. In the "Audio" folder in the Project window, drag and drop the "switch_deactivation" onto the "Audio Clip" properties.
j. Go to "Scripts/AlarmSystems" and drag the "LaserSwitchDeactivation" script to the "prop_switchUnit" object to deactivate the laser.
k. In the "Materials" folder, drag and drop the "prop_switchUnit_screen_unlocked_mat" material to the "Unlocked Mat" in the Inspector.
l. Rename the original "prop_switchUnit" to "prop_switchUnit_bk" in the Prefabs folder and drag and drop the "prop_switchUnit" in the hierarchy to the Prefabs folder to save it as a prefab. Rename the new Prefab to "prop_switchUnit_lab1a"
m. Rename the first switch to "prop_switchUnit_001".
n. Duplicate it THREE times, rename them and set the positions as below.

	Position
prop_switchUnit_002	X = -1.6, Y = 0, Z = 11.3
prop_switchUnit_003	X = -17.7, Y = 0, Z = 33.3
prop_switchUnit_004	X = -30, Y = 0, Z = 33.3

o. Select “prop_switchUnit_001”, drag and drop “fx_laserFence_lasers_001” to the “Laser” in the “Laser Switch Deactivation (Script)”.



p. Repeat the above steps to create the “prop_switchUnit_002 ~ 004”.

q. Save the Scene and the project.

Checkpoint 2: Demo to the instructor or teaching assistant

4. Exercises

Modify the game to fulfil the following requirements:

1. Create a “Sphere” as a Player character with the tag “Player”.
2. The camera follows the player whenever the player moves.
3. Every CCTV camera rotates between 0^o to 60^o bouncingly around the Y axis.
4. When a CCTV camera spots the player, the CCTV camera points to the player instead of playing its animation until the player moves outside the detection range.
5. If the CCTV camera spots the player for about 3 seconds, the game will be restarted.

Checkpoint 3: Demo to the instructor or teaching assistant/take a video with explanations and upload it to Blackboard

Describe and explain the methods you used to achieve the results in your lab report. Any necessary scripts you have added or modified should be included.

5. References

[1] Unity - Game Engine, http://unity3d.com/

[2] Unity - Stealth tutorial, https://www.youtube.com/playlist?list=PLX2vGYjWbI0QGyfO8PKY1pC8xcRb0X-nP

[3] Unity Manual, http://docs.unity3d.com/Manual/index.html

[4] Unity - Script API, http://docs.unity3d.com/ScriptReference/index.html