

1. Objective

In this laboratory exercise, you will use Unity to build a third-person stealth game for Android. You must develop Enemy, Enemy animator control and Enemy AI for the stealth game. The following topics you will learn in this laboratory.

1. Enemy Setup
2. Enemy Animator Controller
3. Enemy Sight
4. Animator Setup
5. Enemy Animation
6. Enemy Shooting
7. Enemy AI

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. Continue with the Lab 1B project, save the Stealth_Lab1B scene to Stealth_Lab2B.

3.1 Enemy Setup

Each enemy will be on a set patrol route by default. If he is awarded the player location by seeing or hearing the player, he should rush to the position of the sighting. If the enemy can see and has a line of sight to the player, he should instead shoot the player.

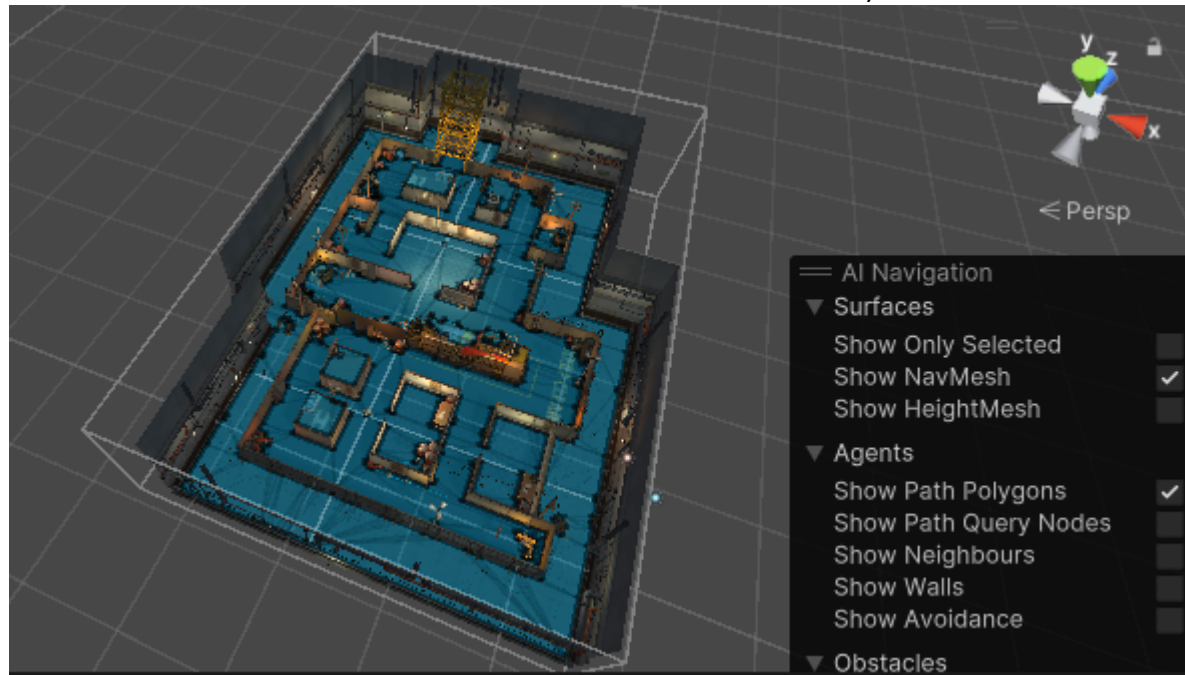
1. To create an enemy

- a. In the “Models” folder, drag and drop “char_robotGuard” to the Scene view.
- b. Set the position to X = -18, Y = 0, Z = 6.5.
- c. Double-click the name to focus on it.
- d. Set the Tag to “Enemy”.
- e. Click Add Component → Sphere Collider.
- f. Set the “Center” to X = 0, Y = 1, Z = 0, the “Radius” to 10, and check “IsTrigger”.
- g. Expand “char_robotGuard” game object and select “char_robotGuard_body”.
- h. Click Add Component → Physics → Capsule Collider.

- i. Set the "Center" to X = 0, Y = 1, Z = 0, the "Radius" to 0.3, and the "Height" to 2.
- j. Select "char_robotGuard".
- k. Click Add Component → Rigidbody, uncheck "Use Gravity" and check "Is Kinematic".
- l. Install AI Navigation System from the Package Manager (Package Manager -> Package: Unity Registry from drop-down menu -> search "AI Navigation System" -> Install).
- m. Click Add Component → Nav Mesh Agent.
- n. Set the "Stopping Distance" to 0.8.

2. Bake the navigation map

- a. Add a GameObject -> AI -> NavMesh Surface to the Hierarchy.
- b. Select "env_stealth_static" in the Hierarchy, click the down arrow that is near the "Static" checkbox and ensure "Navigation Static" is checked.
-make sure debris, ground, liftshaft, props, walls_boundary, and walls_inner are "Static"
- c. Go to "Windows" -> "AI" -> click the "Navigation" tab.
- d. Switch to the "Agents" tab.
- e. Set the "Agent Radius" to 0.3, and the "Agent Height" to 1.
- f. Click the "Bake" button in "NavMesh Surface" in the Hierarchy.



- g. Save the scene and the project.

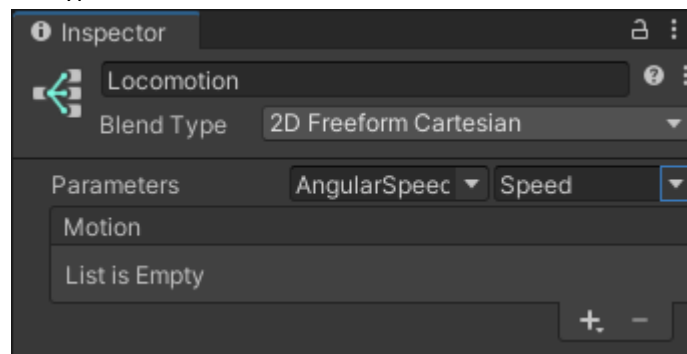
3.2 Enemy Animator Controller

The animator controller for the enemies will be based on the velocity of the nav mesh agent.

1. To make an animated controller for the enemies

- a. Go to the "Animators" folder and rename the original animator of "EnemyAnimator" to "EnemyAnimator_bk"
- b. In the "Animators" folder, click Create → Animator Controller and name it "EnemyAnimator".
- c. Double-click "EnemyAnimator" to open the Animator window.

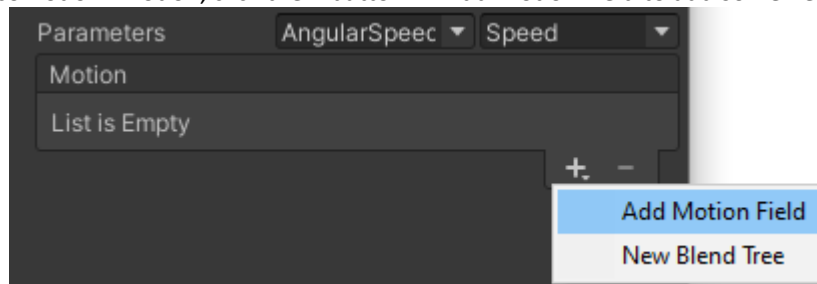
- e. Create the following variables by clicking the “+” button to the right of the “Parameters”.
 - i. AngularSpeed Float
 - ii. Speed Float
 - iii. Shot Float
 - iv. AimWeight Float
 - v. PlayerInSight Bool
- f. Right-click the space on the “Animator” window and select Create State → From New Blend Tree.
- g. Select “Blend Tree” and name it “Locomotion” in the Inspector.
- h. Check the “Foot IK”.
Foot IK = foot planting, a technique used to correct character legs and ankles procedurally at runtime so that the feet can touch the ground.
- i. Double-click click “Locomotion” motion and change the blend tree’s name to “Locomotion”.
- j. Set the “Blend Type” to “2D Freeform Cartesian” and the “Parameters” as follow.



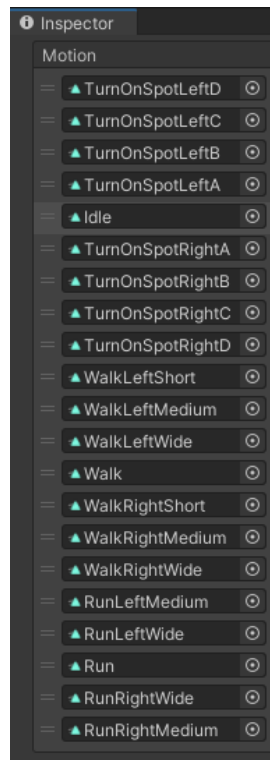
2D Freeform Cartesian is best used when your motions do not represent different directions. With Freeform Cartesian, your X and Y parameters can represent different concepts, such as angular and linear speeds. <https://docs.unity3d.com/Manual/BlendTree-2DBlending.html>

(The x-axis will be represented by the “AngularSpeed” parameter, and the y-axis by “Speed”.)

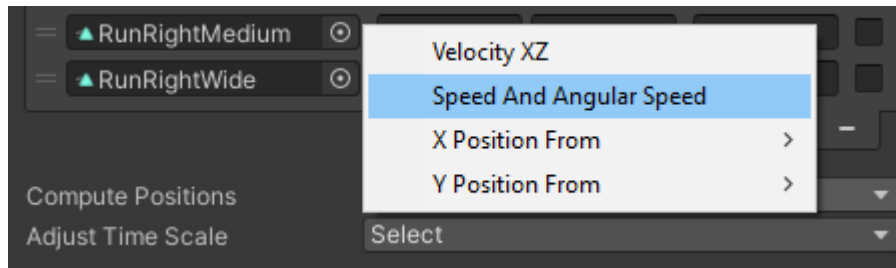
- k. Open the “Animations” folder → Humanoid folder, locate “humanoid_turnOnSpot”, “humanoid_idle”, etc.
- l. Select “Locomotion” motion, click the + button → Add Motion Field to add some fields.



- m. Drag and drop the following motions to the fields.

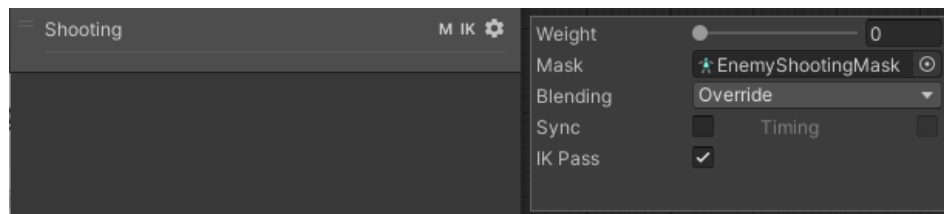


- n. Go to “Compute Position” and select “Speed And Angular Speed” for computation of the position of the animation on the 2D plane.





2. To make the layer for shooting:

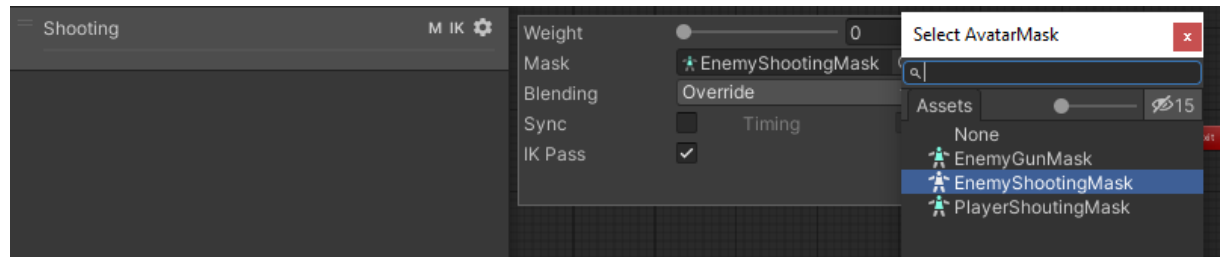
- a. Go to the “Animator” window, and create a new layer called “Shooting”.
- b. Check the “IK Pass” for the Shooting layer.



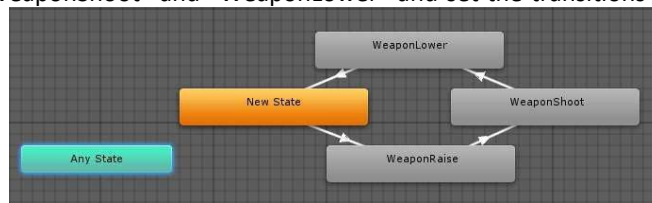
When you want a character to touch an object at a point the user selects or plant its feet convincingly on an uneven surface.

(Inverse kinematics (IK) for Humanoid characters)


- c. Apply the mask to the “Shooting” layer by clicking the  icon → click  button of “Mask” and choose “EnemyShootingMask”.

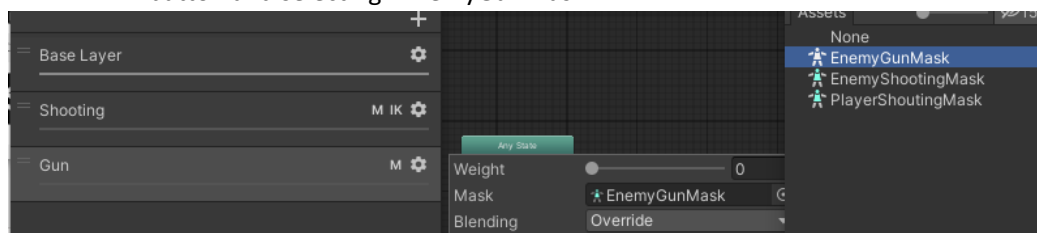


- d. Right-click the “Animator” window and Create State → Empty.
- e. Open the Animations\Humanoid folder in the Project windows and expand “humanoid_weapon_raise”, and drag and drop “WeaponRaise” to the “Animator” window.
- f. Change the “Speed” to 3.
- g. Click “New State” and transition to “WeaponRaise”.
- h. Select the “Transition” and set the condition to “PlayerInSight” is true.
- i. Uncheck “Has Exit Time” and expand Settings to uncheck “Fixed Duration”.
- j. Drag and drop “WeaponShoot” and “WeaponLower” and set the transitions below.

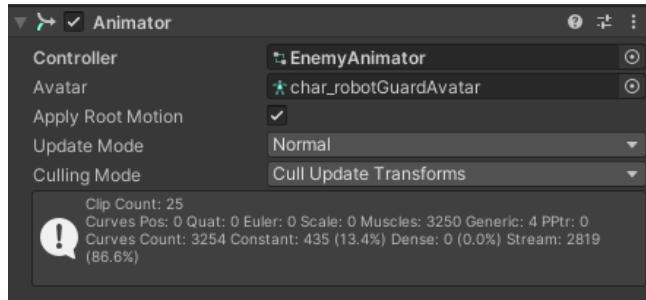


Transition	Condition	Has Exit Time	Fixed Duration
WeaponRaise → WeaponShoot	-none-	Checked	Unchecked
WeaponShoot → WeaponLower	PlayerInSight is false	Unchecked	Unchecked
WeaponLower → New State	-none-	Checked	Unchecked

- k. Create a new animator layer called “Gun” and apply “EnemyGunMask” to the Gun layer by clicking  button and selecting “EnemyGunMask”.



- l. Go to the “Models” folder in the Project windows and expand “char_robotGuard” and drag and drop the “Gripped” motion to the Animator window.
- m. Select “char_robotGuard” in the Hierarchy.
- n. Go to the “Animators” folder, drag and drop “EnemyAnimator” to the “Controller” properties in the Animator component in the Inspector.



- o. Save the scene and the project.

3.3 Enemy Sight

1. Manage the enemy's detection of the player

- a. Go to Scripts\Enemy folder in the Project windows and select "EnemySight" script to apply to "char_robotGuard".

3.4 Enemy Animation

1. Control the movement of the enemy, the enemy will be driven by the animator.

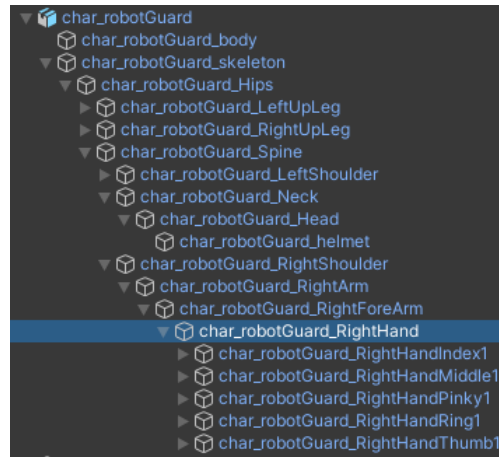
- a. Go to Scripts\Enemy folder in the Project windows and select "EnemyAnimation" script to apply to "char_robotGuard".
- b. Save the scene and the project.


Checkpoint 1: Demo to the instructor or teaching assistant

3.5 Enemy Shooting

1. To create a gun for the enemy:

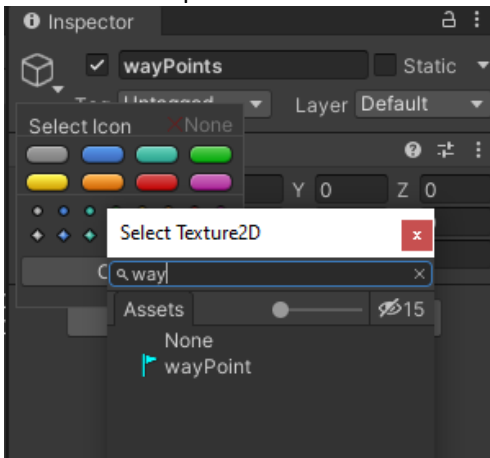
- a. Drill down “char_robotGuard” in the Hierarchy and locate its right hand “char_robotGuard_RightHand”.



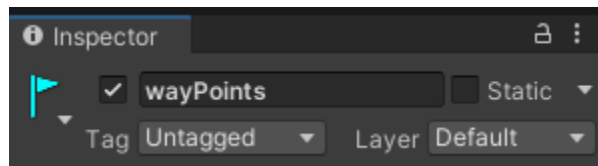
- b. Go to the “Models” folder in the Project windows and drag and drop “prop_sciFiGun_low” to “char_robotGuard_RightHand”.
- c. Select the “prop_sciFiGun_low” in the Hierarchy and double-click the name to focus on it.
- d. Set the position to X = 0.1, Y = 0.021, Z = -0.02 and the rotation to X = 332, Y = 101.4, Z = 263.8.
- e. Create an “empty” GameObject called “fx_laserShot” and make it a child of the “prop_sciFiGun_low”.
- f. Reset the position of “fx_laserShot” and set position to X = 0, Y = 0.05, Z = 0.2.
- g. Click Add Component → Light.
- h. Click Add Component → Line Renderer.
- i. Search for “tile_laser_fx” in the Project window and then drag it to the “Element 0” of “Materials” of “Line Renderer” in the Inspector.
- j. Drag the red dot  from the chart and set the value of Width to **0.05** in the “Line Renderer” component in the Inspector.
- k. Right-click the red line on the chart and click “Add Key”.
- l. Right-click the red dot, click “edit key,” and set the value to 0.025.
- m. Go to the “Light” component and change the “Intensity” to 0.
- n. Go to the “Scripts\Enemy” folder in the Project windows and apply “EnemyShooting” to the “char_robotGuard” object in the Hierarchy.
- o. Go to the “Audio” folder and drag and drop “weapon_sciFiGun_Fire” to the “Shot Clip” variable of “Enemy Shooting (Script)” in the Inspector.
- p. Save the scene and the project.

3.6 Enemy AI

1. The last script that acts as the enemy's brain is "Enemy AI". It makes decisions based on the information given by other scripts.
 - a. Go to the "Scripts\Enemy" folder in the Project windows and apply "EnemyAI" to the "char_robotGuard" object in the Hierarchy.
 - b. Rename the original "char_robotGuard" to "char_robotGuard_bk" from Prefab's folder and drag and drop "char_robotGuard" in the Hierarchy to the Prefabs folder. Rename the new Prefab to "char_robotGuard_lab2b".
 - c. Duplicate 2 more "char_robotGuard" by pressing Ctrl + D. Rename the "char_robotGuard" to "char_robotGuard_001", "char_robotGuard_002" and "char_robotGuard_003".
 - d. Set the position of "char_robotGuard_002" to X = -19, Y = 0, Z = 37.
 - e. Set the position of "char_robotGuard_003" to X = -29, Y = 0, Z = 43.5.
2. To create waypoints for the enemies to patrol
 - a. Create an empty GameObject called "wayPoints" and reset its position to X = 0, Y = 0, and Z = 0.
 - b. Duplicate the "wayPoints" game object and name it "wayPoint_001". Make it a child of "wayPoints".
 - c. Click the icon on the top left corner in the Inspector.



- d. Click "Other...", and search for "waypoint". Select and apply it.

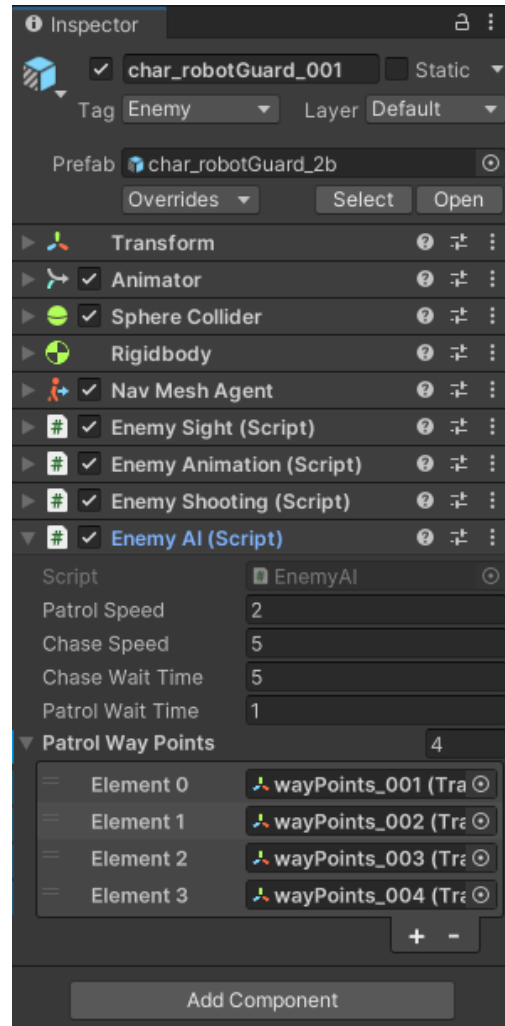


- e. Duplicate "wayPoint_001" 11 times and rename them with IDs from 002 to 011.
- f. Set their positions as below:

Way Point	Position
wayPoint_001	X= -26.75, Y = 0, Z = 7
wayPoint_002	X= -25.4, Y = 0, Z = 14.5
wayPoint_003	X= -18.8, Y = 0, Z = 14
wayPoint_004	X= -18.5, Y = 0, Z = 7
wayPoint_005	X= -10, Y = 0, Z = 36.4
wayPoint_006	X= -10, Y = 0, Z = 44.8
wayPoint_007	X= -17.9, Y = 0, Z = 44.8
wayPoint_008	X= -17.8, Y = 0, Z = 36.4
wayPoint_009	X= -27, Y = 0, Z = 35.4

wayPoint_010	X= -17.9, Y = 0, Z = 43
wayPoint_011	X= -28.2, Y = 0, Z = 43

- g. Select “char_robotGuard_001”, and drag and drop “way point 1, 2, 3, 4” to the “Patrol Way Points” variable of the Enemy AI (Script) in the Inspector.



- h. Select “char_robotGuard_002”, and drag and drop the “way point 5, 6, 7, 8” to the “Patrol Way Points” in the “Enemy AI (Script)” in the Inspector.
- i. Select “char_robotGuard_003”, and drag and drop the “way point 8, 9, 10, 11” to the “Patrol Way Points” in the “Enemy AI (Script)” in the Inspector.
- j. Save the scene and the project and press the Play button to test the game.

Checkpoint 2: Demo to the instructor or teaching assistant

4. Exercises

Modify the game to fulfil the following requirements:

1. Mimi stops following the player when “Enemy” is within 3 units in radius and disappears for a period of time, such as 5 seconds.
2. After Mimi reappears, if the “Enemy” is still within 3 units of radius, Mimi stops (idle). If the “Enemy” is outside the 3 units of radius, Mini follows the player again.

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 include.

3. References

- [1] Unity - Game Engine, <http://unity3d.com/>
- [2] Unity - Stealth tutorial, <https://www.youtube.com/playlist?list=PLX2vGYjWbl0QGyfO8PKY1pC8xcRb0X-nP>
- [3] Unity Manual, <http://docs.unity3d.com/Manual/index.html>
- [4] Unity - Script API, <http://docs.unity3d.com/ScriptReference/index.html>
- [5] Unity - The Particle System, <https://learn.unity.com/project/creative-core-vfx>
- [6] Unity – Tutorials, <https://learn.unity.com/tutorials>