Candice: AI for Games

User Manual

Contents

Getting Started

In this system, there are three main parts.

- Candice AI Manager
    - This houses all the higher level logic, such as calculating shortest paths using Pathfinding algorithms.
    - It also handles the registration of AI agents and issues each agent a unique Agent ID.
    - An instance of Candice AI Manager is required for **Pathfinding**, **Obstacle Avoidance** and **Object Detection** functionality.
- Candice AI Controller
    - This script controls the actual AI agents. It supports Finite State Machines and Behaviour tress and even includes an attack and Damage System.
- Simple AI Controller
    - This basic AI logic such as moving from one point to another, destroying on collision etc. It is best used for bullets or missile type systems. Normally, it would be attached to the projectiles fired by the AI Agent.

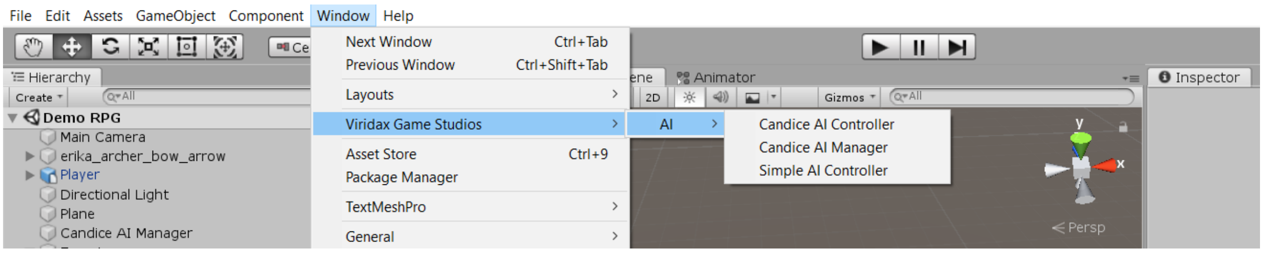
Candice AI Controller

Before you begin using the AI Controller script, make sure you have done the following first:

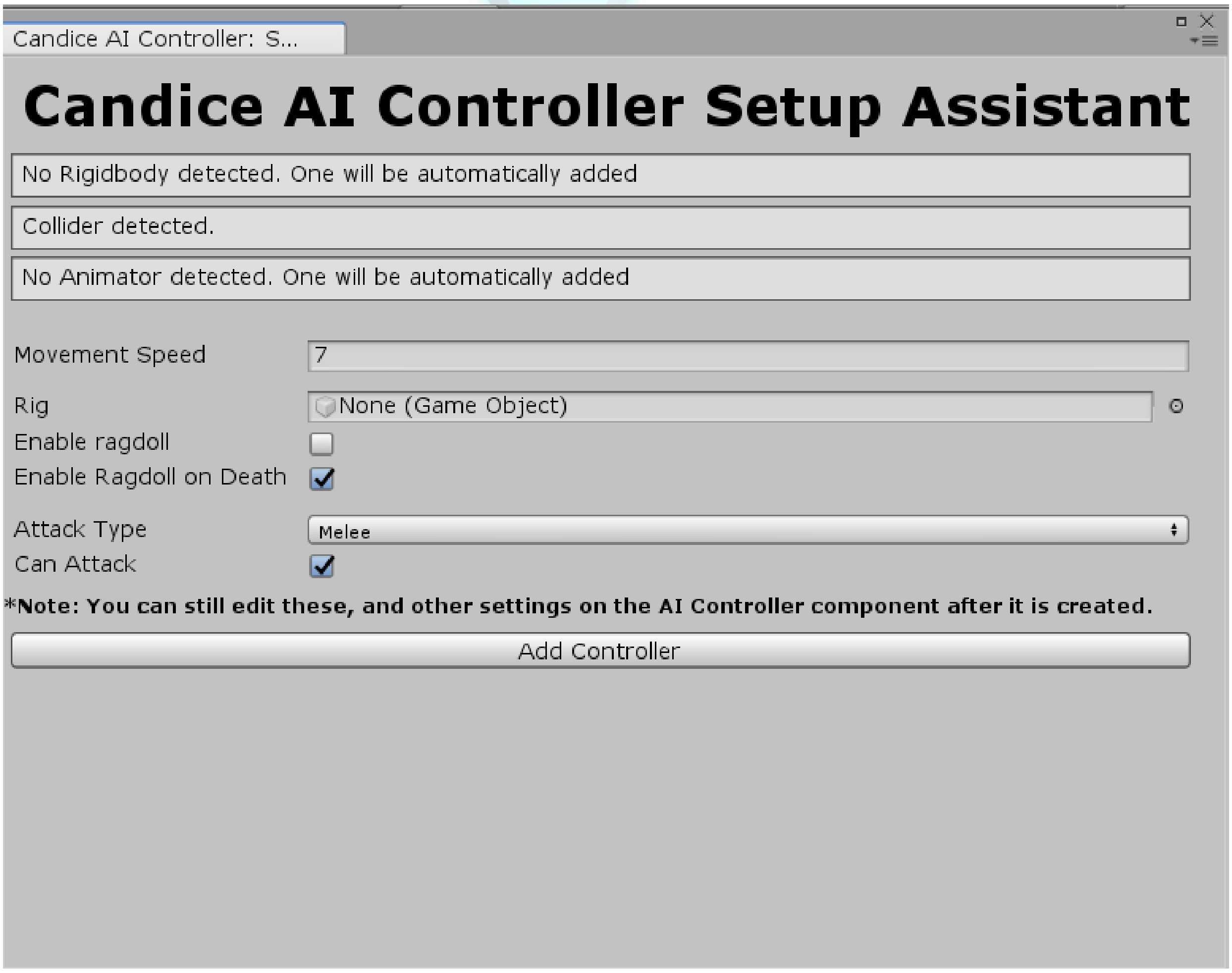
1. Ensure you have animations for Idle, Running (and/or walking), receiving damage and attacking. Idle should be the default animation state.
2. Note: The it will still work without animations, however it will require a bit more setting up to get the Attack/Damage system working
3. Set the triggers and Boolean conditions accordingly:
    a. You can use the conditions in the script or use your own.
    b. If you use your own conditions, you will have to change them in the script as well. (Where it says anim.SetBool() and anim.SetTrigger())

Adding the Components

In order to add the controller, click on **Window -> Viridax Game Studios -> AI -> Candice AI Manager Window -> Viridax Game Studios -> AI -> Candice AI Controller**, or **Window -> Viridax Game Studios -> AI -> Simple AI Controller**.



When adding the AI Controller, a Setup Assistant will popup, allowing you to initialise the agent with some basic variables. It will also automatically add a Rigidbody, Collider and Animator if they are not found.



Candice AI Controller Setup Assistant

No Rigidbody detected. One will be automatically added

Collider detected.

No Animator detected. One will be automatically added

Movement Speed: 7

Rig: None (Game Object)

Enable ragdoll: □

Enable Ragdoll on Death: ☑

Attack Type: Melee

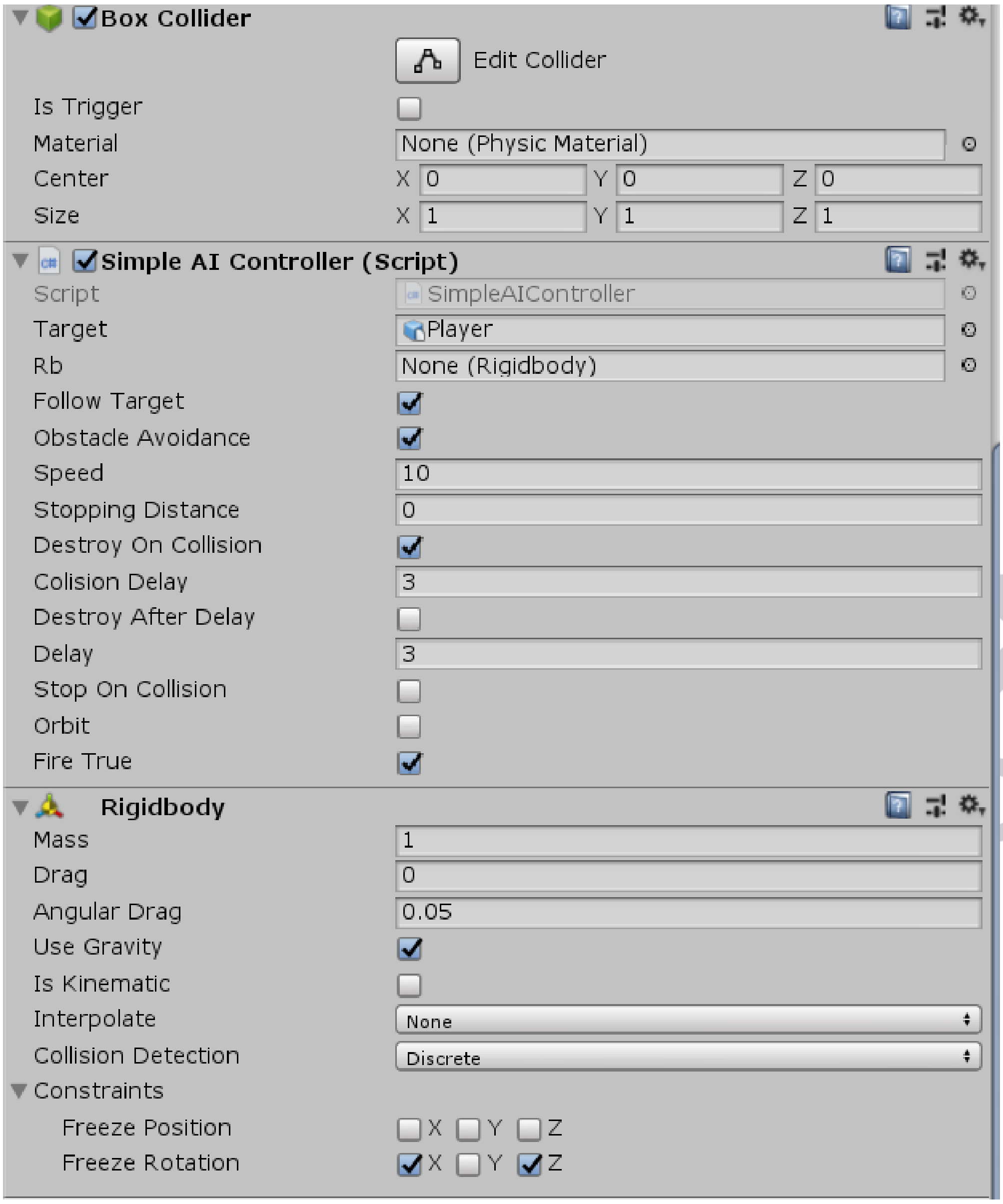
Can Attack: ☑

*Note: You can still edit these, and other settings on the AI Controller component after it is created.

Using the Controllers

Simple AI Controller

This script is straight-forward and easy to use. Once the script has been added to the game object, assign a target and it will automatically move to the target. Any game object can be assigned as a target.



☑ Box Collider

Is Trigger ☐
Material: None (Physic Material)
Center: X 0 Y 0 Z 0
Size: X 1 Y 1 Z 1

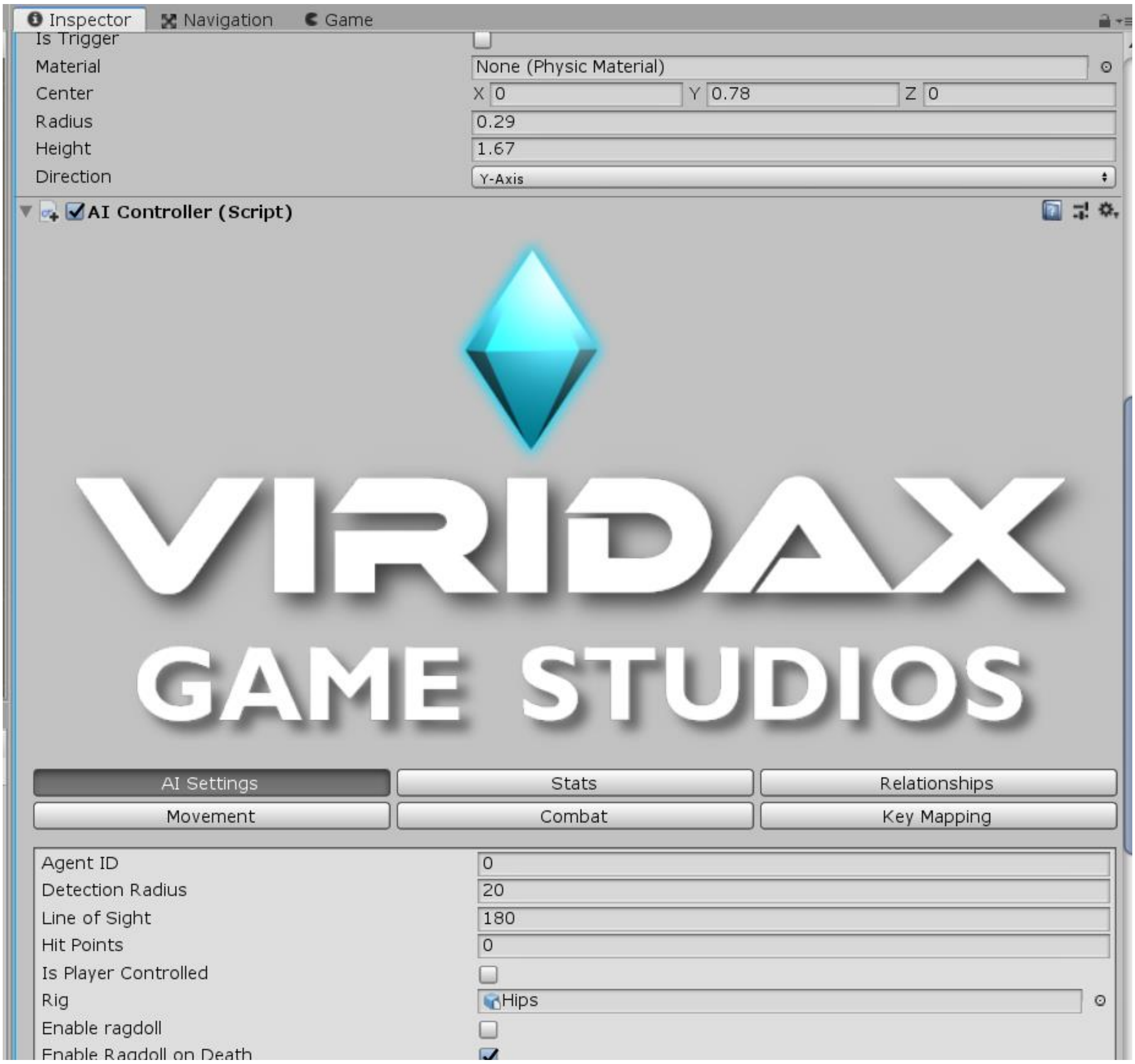
☑ Simple AI Controller (Script)

Script: SimpleAIController
Target: Player
Rb: None (Rigidbody)
Follow Target ☑
Obstacle Avoidance ☑
Speed: 10
Stopping Distance: 0
Destroy On Collision ☑
Colision Delay: 3
Destroy After Delay ☐
Delay: 3
Stop On Collision ☐
Orbit ☐
Fire True ☑

Rigidbody

Mass: 1
Drag: 0
Angular Drag: 0.05
Use Gravity ☑
Is Kinematic ☐
Interpolate: None
Collision Detection: Discrete
Constraints
Freeze Position ☐ X ☐ Y ☐ Z
Freeze Rotation ☑ X ☐ Y ☑ Z

Candice AI Controller



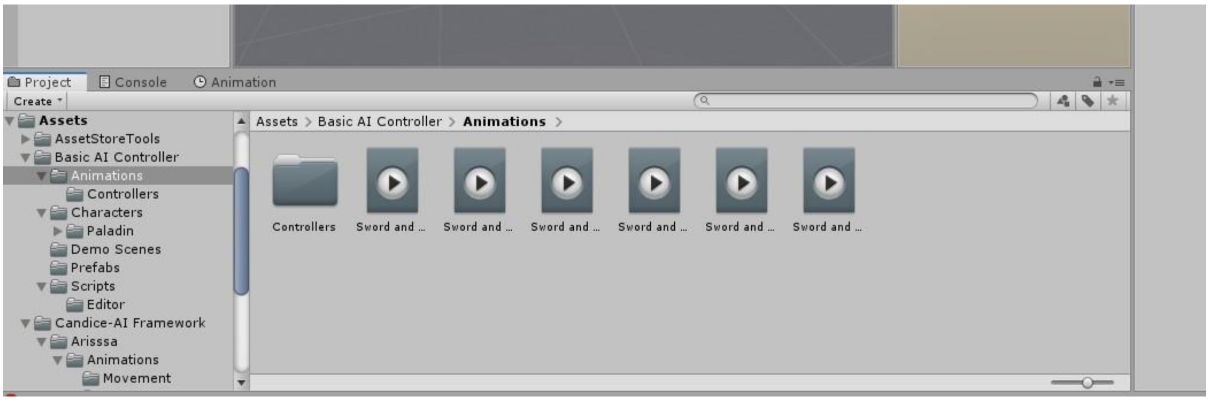
Is Trigger □
Material None (Physic Material)
Center X 0 Y 0.78 Z 0
Radius 0.29
Height 1.67
Direction Y-Axis

Agent ID 0
Detection Radius 20
Line of Sight 180
Hit Points 0
Is Player Controlled □
Rig Hips
Enable ragdoll □
Enable Ragdoll on Death ☑

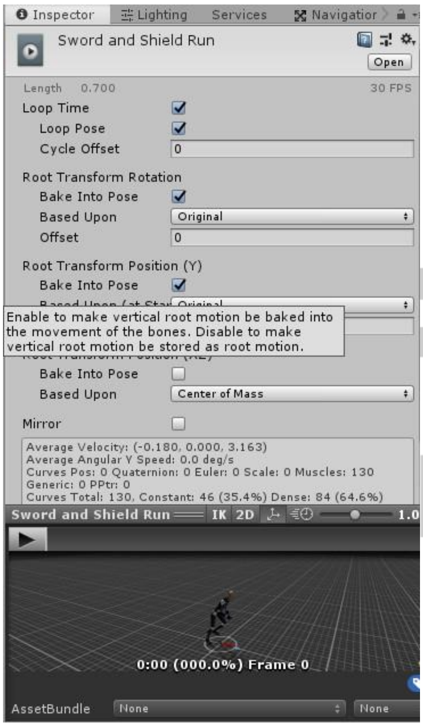
The AI Controller works best with animations. However, if you do not have animations, uncheck **Can Attack** in the inspector (int eh Combat tab) in order to prevent a bug when it tries to attack.

***Note**: you will require some basic knowledge on how to set up animator controllers with animations.

To use the script with animations, ensure you have animations for Idle, Walk, Run, Dame and Attack.



Make sure walking and running animations have “Loop Time” and “Loop Pose” enabled in order for the animations to loop seamlessly. Also, depending on what animations you use, you might have to check some other settings and also ensure that the 3D model is using the correct rig (usually it is Humanoid).



Sword and Shield Run

Length 0.700 30 FPS

Loop Time ☑

Loop Pose ☑

Cycle Offset 0

Root Transform Rotation

Bake Into Pose ☑

Based Upon Original

Offset 0

Root Transform Position (Y)

Bake Into Pose ☑

Based Upon (at Star Original

Enable to make vertical root motion be baked into the movement of the bones. Disable to make vertical root motion be stored as root motion.

Root Transform Position (XZ)

Bake Into Pose □

Based Upon Center of Mass

Mirror □

```
Average Velocity: (-0.180, 0.000, 3.163)
Average Angular Y Speed: 0.0 deg/s
Curves Pos: 0 Quaternion: 0 Euler: 0 Scale: 0 Muscles: 130
Generic: 0 PPtr: 0
Curves Total: 130, Constant: 46 (35.4%) Dense: 84 (64.6%)
```

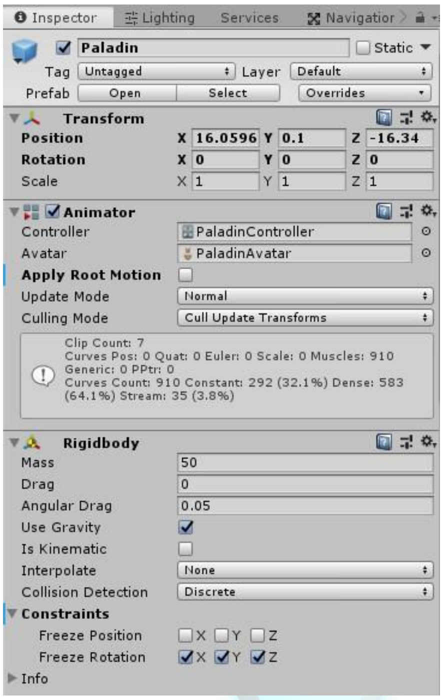
Sword and Shield Run

0:00 (000.0%) Frame 0

When you add the script from the Menu, it will automatically add a Rigidbody, a Collider and Animator. After that, follow these steps:

1. In the Animator component, uncheck “Apply Root Motion.”
2. In the Rigidbody, ensure that “Use Gravity” is checked.
3. Also in the Rigidbody go to the constraints section and check Freeze Rotation for X and Z.
4. In the Collider, adjust the height, radius and centre according to your character.
5. These settings are just a guideline and you can play around with them to suit your situation.

Note: The Setup Assistant does these things automatically.



☑ Paladin □ Static

Tag Untagged Layer Default

Prefab

Transform

Position	X	16.0596	Y	0.1	Z	-16.34
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

☑ **Animator**

Controller	PaladinController
Avatar	PaladinAvatar
Apply Root Motion	□
Update Mode	Normal
Culling Mode	Cull Update Transforms

Clip Count: 7
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 910
Generic: 0 PPtr: 0
Curves Count: 910 Constant: 292 (32.1%) Dense: 583 (64.1%) Stream: 35 (3.8%)

Rigidbody

Mass	50
Drag	0
Angular Drag	0.05
Use Gravity	☑
Is Kinematic	□
Interpolate	None
Collision Detection	Discrete

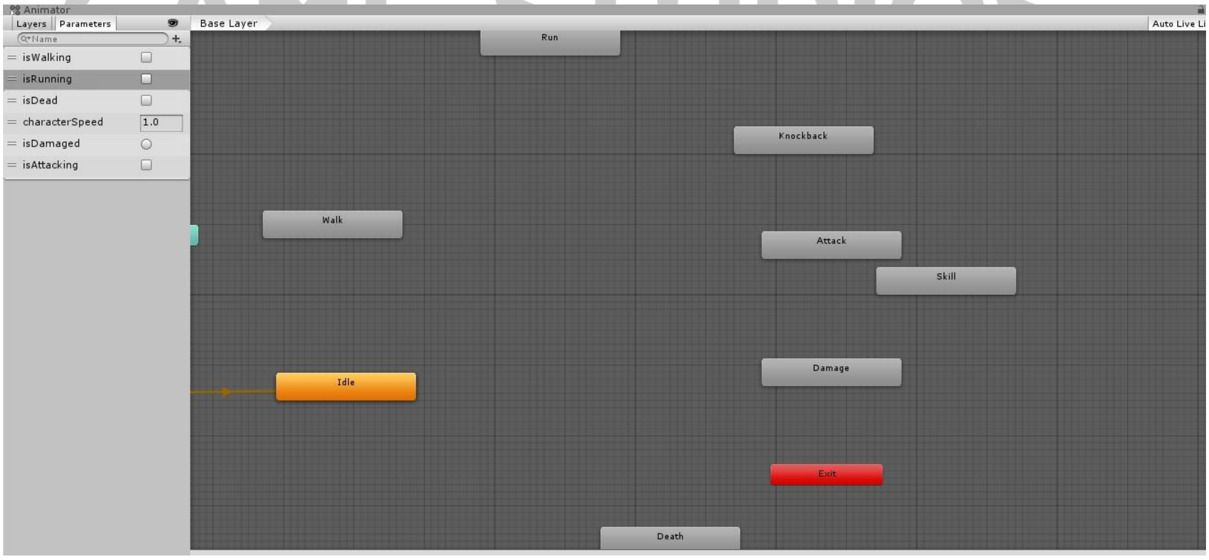
Constraints

Freeze Position	□ X □ Y □ Z
Freeze Rotation	☑ X ☑ Y ☑ Z

Info

Once all of these have been set up, we can now set up the animations. This animation controller set up works with the default Finite State Machine implementation.

1. Create a new Animator Controller and add it to the Controller section in the Animator. In the case of the demo scene it is **Paladin Controller**.
2. Double click on your Animator Controller and then drag all the required animations into it.
3. By now you should have something looking similar to this:

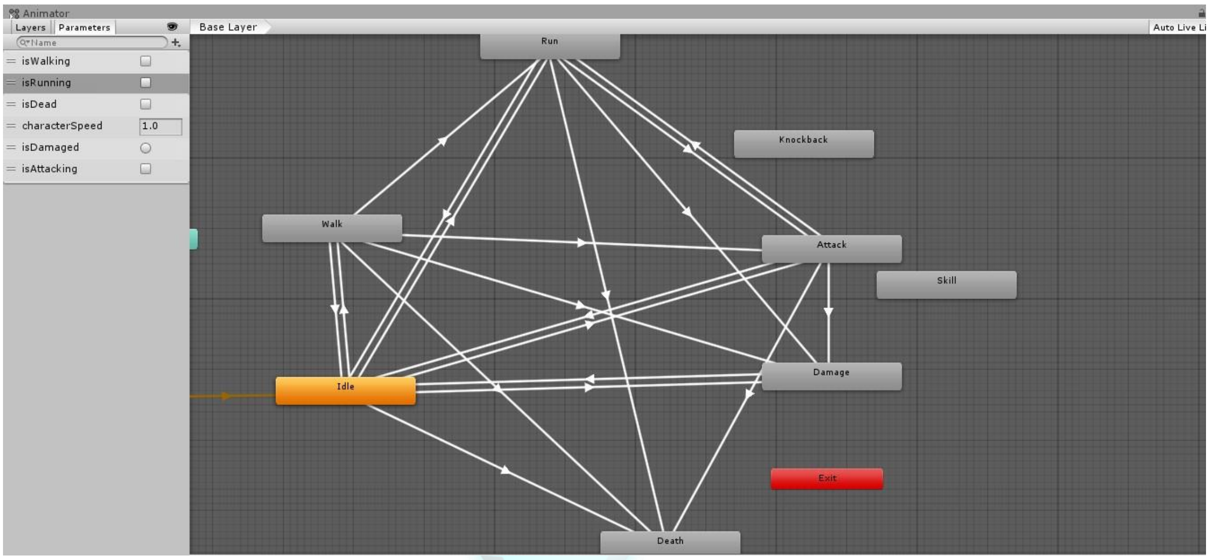


isWalking	□
isRunning	□
isDead	□
characterSpeed	1.0
isDamaged	○
isAttacking	□

4. You can now add transitions between the different animations. You will need the following conditions:
   a. isWalking (Boolean), isRunning (Boolean), isDead (Boolean), isDamaged (Trigger), isAttacking (Boolean).

b. The demo scene will show you where each transition needs to be and with what condition. Also take note that some transitions have “Has Exit Time” unchecked.

5. Your animator controller should look something like this by now:



isWalking □
isRunning □
isDead □
characterSpeed 1.0
isDamaged ○
isAttacking □

***Note: The AI Controller uses Behaviour Tress by default. So you will have to set up one of you own and attach it to the Game Object. A more detailed explanation can be found under the features section.**

Features

Character Class

The Basic AI Controller inherits from a class called “Character”. This class stores information/attributes about the character such as Strength, Intelligence, Faith, Attack Damage and more. It also keeps track of the Player Level.

Character Stats classes

There are several statistics but all of them are controlled by 3 primary attributes:

- Strength.
- Intelligence.
- Faith.

Each of these attributes directly affect three forms of damage.

- Physical Damage (Strength)
- Magical Damage (Intelligence)
- Spiritual Damage (Faith)

Levelling Up

The Character Stats class supports levelling up by calling the LevelUp() method. This will automatically adjust the character attributes. This feature enables the AI characters to level up and become stronger as the game progresses.

Finite State Machines

The controller supports different states that the character can be in, using Finite State Machines. The states are

- Idle
- Follow
- Attack
- Patrol

By default, the controller should change between states automatically. More states can be easily added.

***Note: The character can only be in one state at a time. Finite state machines are the default implementation if the agent is controlled by the player.**

In order for a state to work, you must create an Action class that inherits from (FSMAction) and then assign it to the state you want. For example, create a MoveAction Action that can be assigned to the Follow State and Patrol State (since they both require the agent to move).

Patrol State

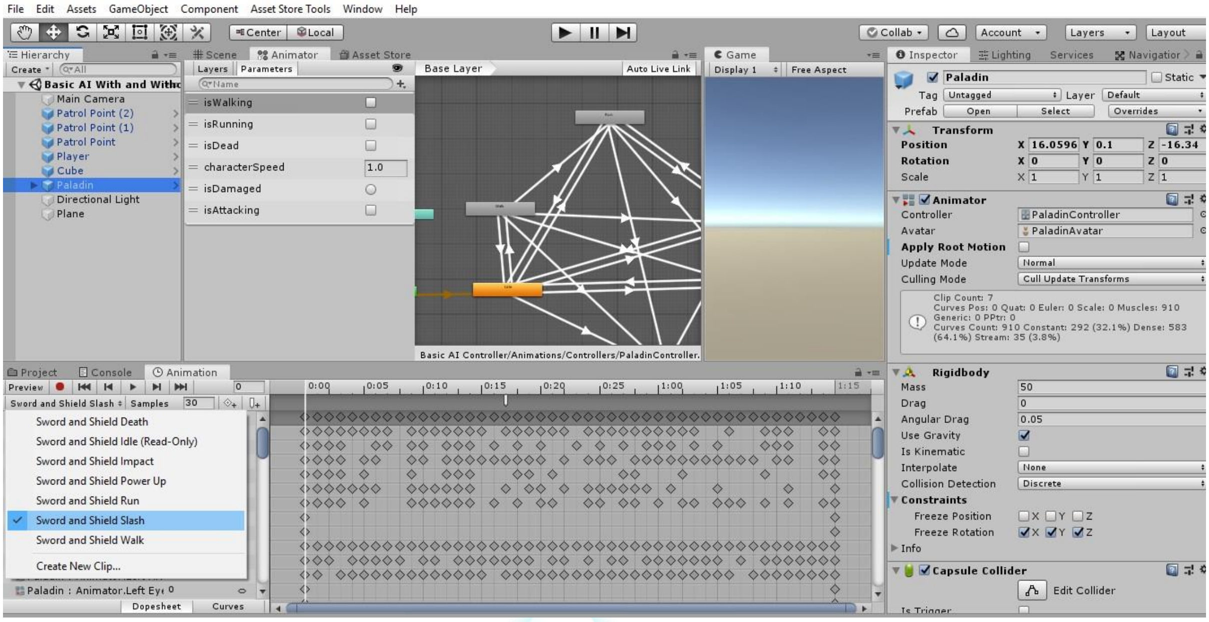
For the character to patrol between several points, add the required Patrol Points to the controller in the inspector:

The patrol points can be anything from a simple empty GameObject to more complex things. Make sure each Patrol Point is tagged as ‘PatrolPoint’.

Attack State

The AI unit can also attack and deal damage to objects around. You will need to use an animation Event to call the Attack() method in the script. To do so, follow these steps:

1. Select the 3D Model, in this case it is **Paladin**.
2. Select the **Animation Window**.
3. Select the Attack animation.
4. Add an animation Event at the point where you feel the damage should be dealt. Usually where the sword comes down and makes contact with the player.
5. Choose Attack() as the function to call.


File Edit Assets GameObject Component Asset Store Tools Window Help
Center Local
Collab Account Layers Layout
Hierarchy
Scene Animator Asset Store
Game
Inspector Lighting Services Navigation
Project Console Animation
Dopesheet Curves

Hierarchy: Basic AI With and Witho
- Main Camera
- Patrol Point (2)
- Patrol Point (1)
- Patrol Point
- Player
- Cube
- Paladin
- Directional Light
- Plane

Parameters:

isWalking	□
isRunning	□
isDead	□
characterSpeed	1.0
isDamaged	○
isAttacking	□

Base Layer — Auto Live Link

Basic AI Controller/Animations/Controllers/PaladinController.

Display 1 — Free Aspect

Inspector: ☑ Paladin □ Static

Tag Untagged — Layer Default

Prefab Open Select Overrides

Transform

	X	Y	Z
Position	16.0596	0.1	-16.34
Rotation	0	0	0
Scale	1	1	1

☑ Animator

Controller	PaladinController
Avatar	PaladinAvatar
Apply Root Motion	□
Update Mode	Normal
Culling Mode	Cull Update Transforms

Clip Count: 7
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 910
Generic: 0 PPtr: 0
Curves Count: 910 Constant: 292 (32.1%) Dense: 583 (64.1%) Stream: 35 (3.8%)

Rigidbody

Mass	50
Drag	0
Angular Drag	0.05
Use Gravity	☑
Is Kinematic	□
Interpolate	None
Collision Detection	Discrete

Constraints
Freeze Position □X □Y □Z
Freeze Rotation ☑X ☑Y ☑Z

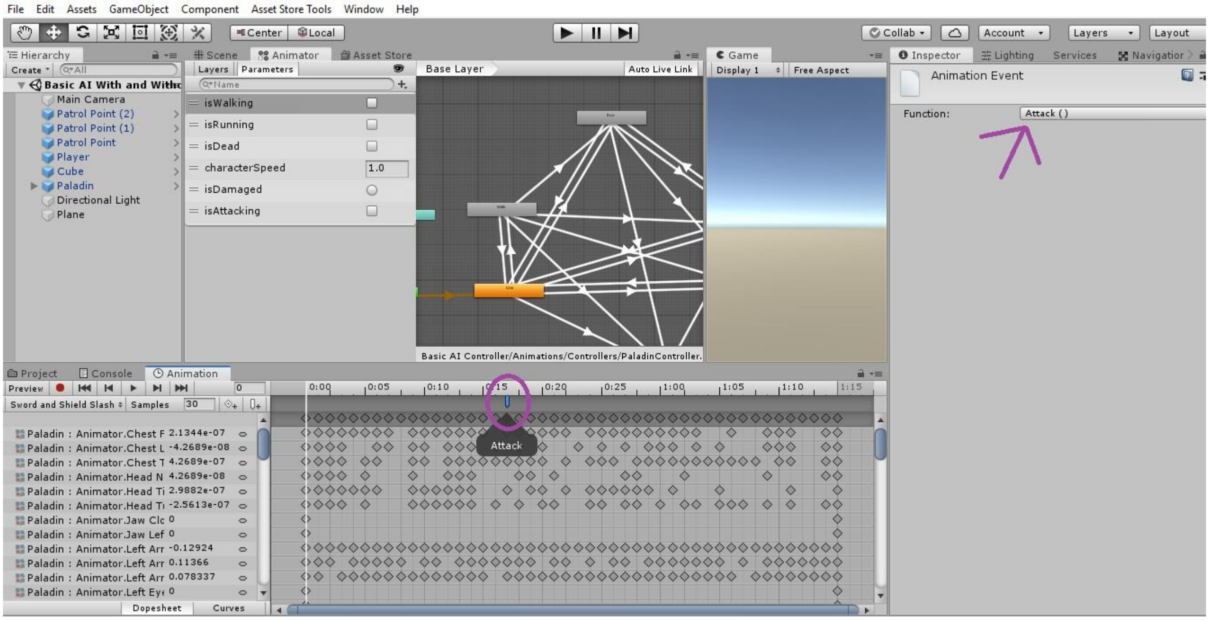
Info

☑ Capsule Collider — Edit Collider — Is Trigger □

Animation: Preview — Sword and Shield Slash — Samples 30

- Sword and Shield Death
- Sword and Shield Idle (Read-Only)
- Sword and Shield Impact
- Sword and Shield Power Up
- Sword and Shield Run
- ✓ Sword and Shield Slash
- Sword and Shield Walk
- Create New Clip...

Paladin : Animator.Left Eye 0


File Edit Assets GameObject Component Asset Store Tools Window Help
Center Local
Collab Account Layers Layout
Hierarchy
Scene Animator Asset Store
Game
Inspector Lighting Services Navigation
Project Console Animation
Attack
Dopesheet Curves

Inspector: Animation Event

Function: Attack ( )

Animation: Preview — Sword and Shield Slash — Samples 30

Paladin : Animator.Chest F	2.1344e-07
Paladin : Animator.Chest L	-4.2689e-08
Paladin : Animator.Chest T	4.2689e-08
Paladin : Animator.Head N	4.2689e-08
Paladin : Animator.Head Ti	2.9882e-07
Paladin : Animator.Head Ti	-2.5613e-07
Paladin : Animator.Jaw Clc	0
Paladin : Animator.Jaw Lef	0
Paladin : Animator.Left Arr	-0.12924
Paladin : Animator.Left Arr	0.11366
Paladin : Animator.Left Arr	0.078337
Paladin : Animator.Left Eye	0

Behaviour Trees

The AI Controller also supports behaviour trees. They are the default implementation if the agent is AI controlled.

You will have to create a script that inherits from MonoBehaviour. In the script, you will have to create your various nodes, starting from the root node. This will require a bit of knowledge on how behaviour trees work. A link is provided at the end explaining Behaviour Trees.

Damage System

The Damage System works by sending a Raycast as a sphere around the character. If it collides with an object marked with any of the Enemy Tags, a message will be sent to the enemy object in order to receive damage. Ensure that the receiving object has a method with the name ReceiveDamage(). It should have the following method signature

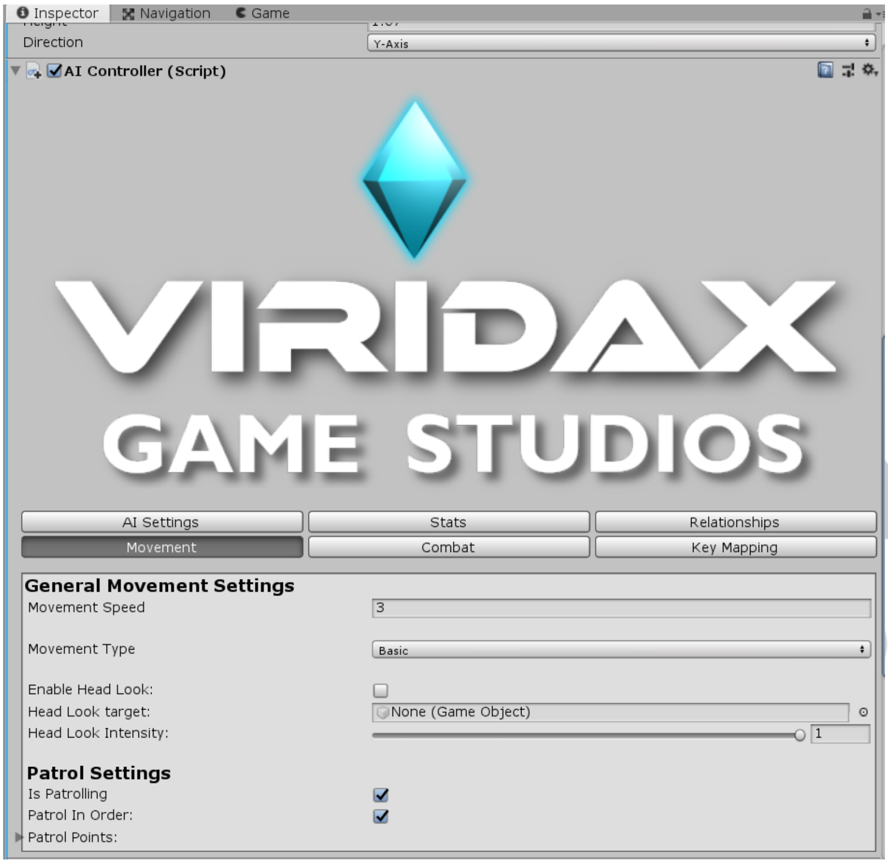
Public void ReceiveDamage(float damage) { }

Head Look

The Basic AI Controller also has a Head Look feature. This allows the character to dynamically look at any object in the game. To use this feature follow these steps:

1. Click on the Movement tab.
2. Check the “Enable Head Look” option.
3. Select a Target, you can drag and drop any gameobject into the slot.

The intensity determines how intensely the character will focus on the object.



Direction: Y-Axis

General Movement Settings

Movement Speed: 3

Movement Type: Basic

Enable Head Look: □

Head Look target: None (Game Object)

Head Look Intensity: 1

Patrol Settings

Is Patrolling: ☑

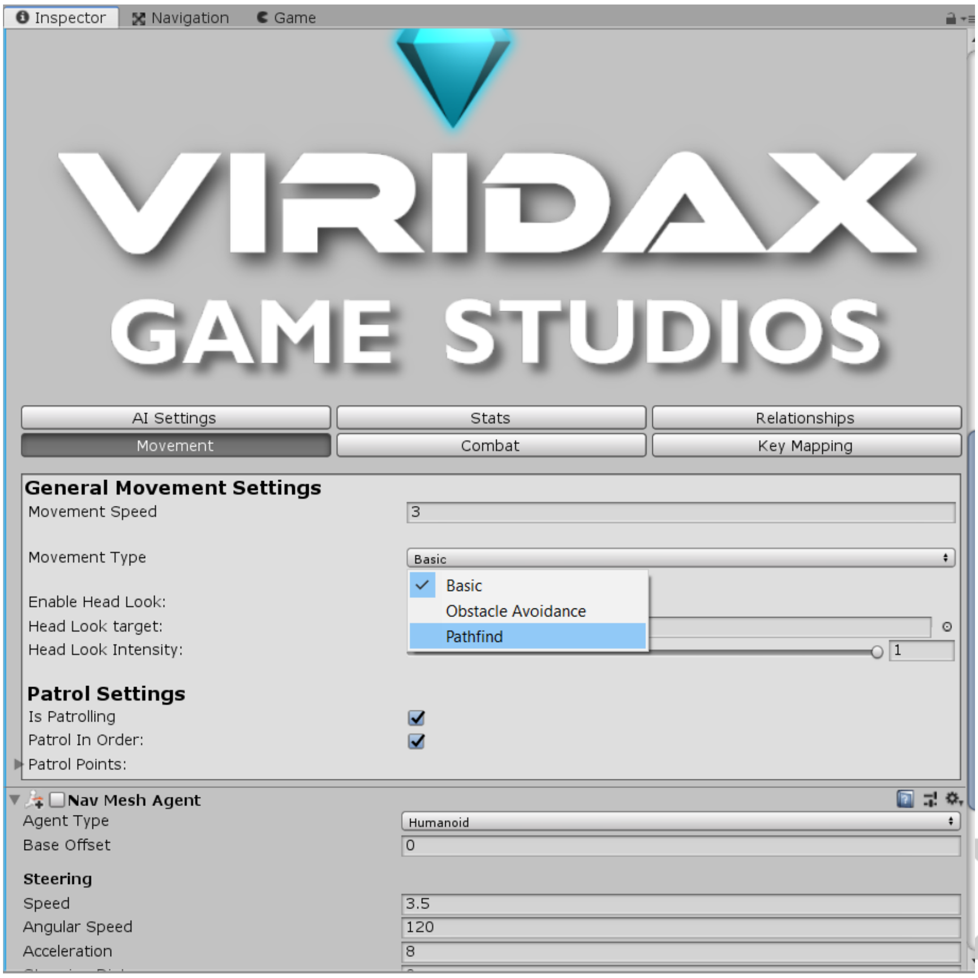
Patrol In Order: ☑

Patrol Points:

Movement

The AI controller supports three types of movement:

- Simple.
- Obstacle Avoidance
  - This evades immediate obstacles.
- Pathfinding
  - This uses the A* algorithm to find the shortest path to the target.



General Movement Settings

Movement Speed: 3

Movement Type: Basic

Enable Head Look:

Head Look target:

Head Look Intensity: 1

Patrol Settings

Is Patrolling ☑

Patrol In Order: ☑

Patrol Points:

□ Nav Mesh Agent

Agent Type: Humanoid

Base Offset: 0

Steering

Speed: 3.5

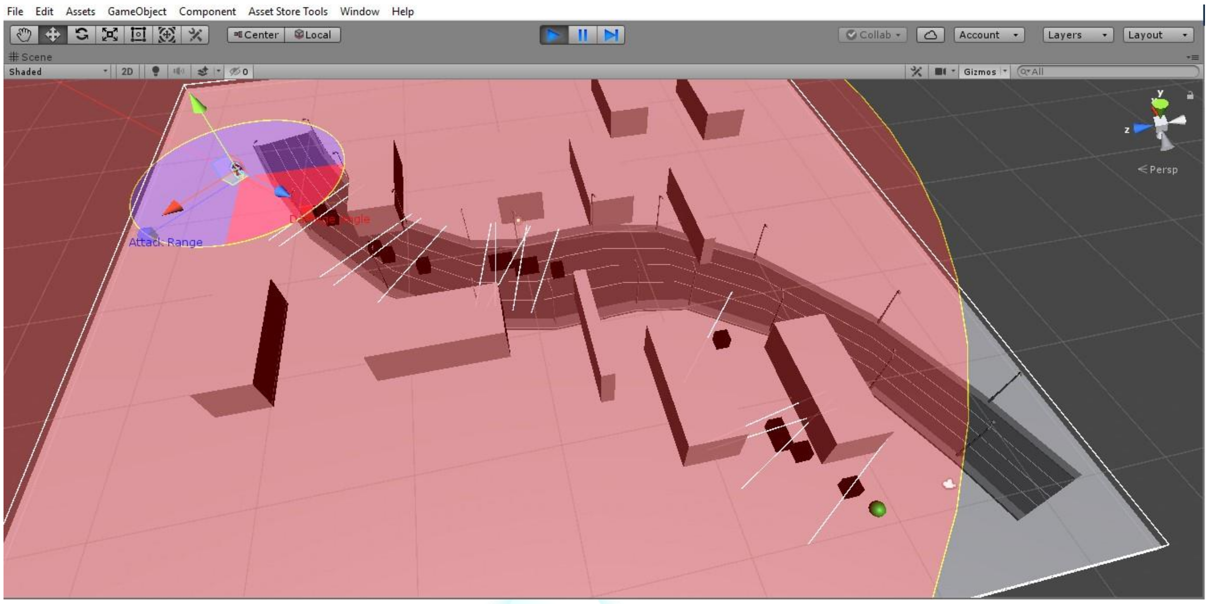
Angular Speed: 120

Acceleration: 8

Obstacle Avoidance

This allows the AI to evade immediate obstacles, within a certain range. This algorithm is most efficient in straight line paths with a few random obstacles. However, it can also be used in more complex scenes, but efficiency is not guaranteed.

A* Pathfinding



This algorithm allows the agent to find the shortest path to the target.

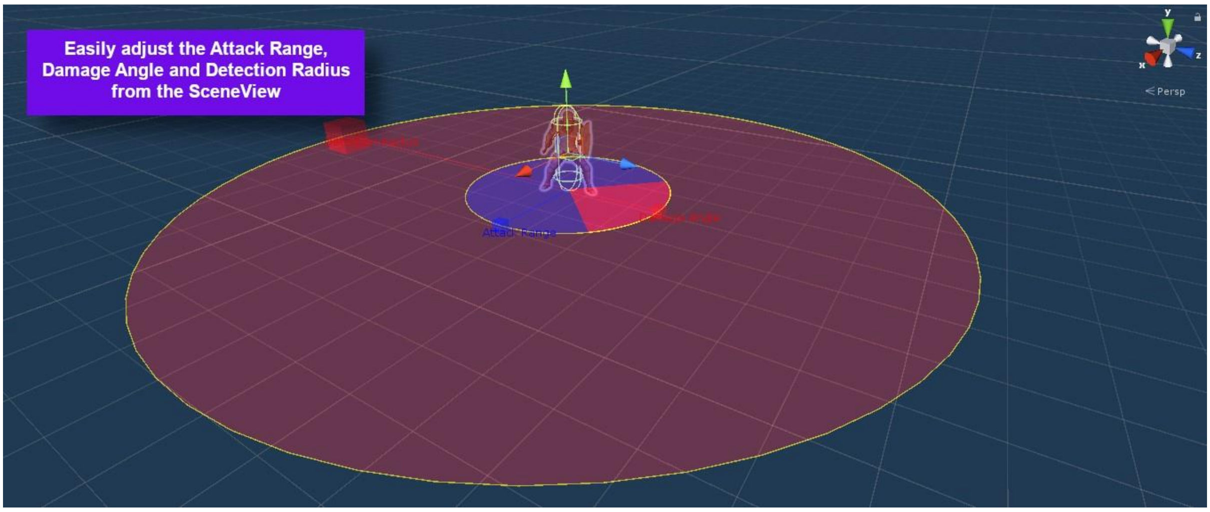
To make it work in your own scenarios, follow these steps:

1. Ensure you have an Candice AI Manager script attached to a GameObject
    a. Click on **Window -> Viridax Game Studios -> AI -> Candice AI Manager and it will automatically create one for you, with a Grid component as well**.
2. Tweak the Grid settings to fit your terrain and define an unwalkable mask. Unwalkable mask is the Layer which the algorithm will define as unwalkable, and thus the AI will never walk on terrain/gameobjects with that Layer. (e.g a layer with the name “Obstacles”)
3. Define Walkable Regions.
    a. For each walkable region, you must specify a Terrain Mask (basically a Layer that will be assigned to a terrain/gameobject) and a movement penalty for that terrain. This allows the AI to “prefer” certain terrain types over others. E.g, it might prefer moving on the road layer as opposed to the sand layer.

Now you can call **StartFindingPath()** and **StopFindingPath()** respectively and it will automatically calculate the best path and move the agent.

Editor and SceneView

Basic A.I Controller also has powerful editor and SceneView controls. You can adjust the properties including attack range and detection radius right from the SceneView. You can also adjust all other properties, like enemy tags, move speed and more from the inspector. There’s no need to touch the script if you don’t want to.


Easily adjust the Attack Range, Damage Angle and Detection Radius from the SceneView
Attack Range
y
x
z
Persp

Support

We at Viridax Game Studios strive to provide the best AAA quality video game script assets to the world. If you have any issues with this asset, suggestions, or just want to send some nice feedback, then feel free to contact us on one of our channels.

Support email: natubeast@gmail.com

Discord Server: https://discord.gg/Tar7nxX

Facebook: https://www.facebook.com/viridaxgamestudios/

I'd also appreciate it if you could support me on Patreon. It will allow me to keep creating awesome Development Tools for free.

Patreon: https://www.patreon.com/natubeast

References

Item	Author	Website
Paladin 3D Model	Mixamo	https://www.mixamo.com
Sword and Shield Animations	Mixamo	https://www.mixamo.com
Erika 3D Model	Mixamo	https://www.mixamo.com