Super Goalie Basic

Andrew Manyore

TABLE OF CONTENTS

INTRODUCTION

Super Goalie Basic is a basic implementation of a football goal keeper artificial intelligence. Basic goal keeper states have been implemented like tend goal, intercept shot and punch ball.

Super Goalie Basic was designed to be integrated with other projects. You can integrate Super Goalie Basic with any of your already existing projects.

SETTING UP

INTRODUCTION

In this section you will learn how to set up Super Goalie Basic. The set-up includes first setting up (creating) the various entities that are used in Super Goalie Basic and then the entire Super Goalie Basic scene

SETTING UP THE ENTITIES

BALL

- Create an empty game object in the scene and name it ball.
- Attach the ball model to it. Make sure the ball sits on this empty game object.
- Attach the ball ***Ball.cs*** script to the Ball object you created in step 1. The ***Ball.cs*** script requires the rigid body component and the sphere collider component. Both components will be added automatically.
- Adjust the rigid body. Make sure it matches the rigid body in the picture.
- Add the ***pMatBall*** physics material to the Sphere Collider material to make the ball bouncy.
- Save the ball as a prefab

GOAL

- Create an empty game object in the scene and name it goal.
- Add the goal script to the goal
- Attach the goal model to the goal.
- Attach a cube to the goal. Name it goal trigger. Set its layer to ***GoalTrigger***. Adjust it so that it covers the entire mouth of the goal model. Add the ***GoalTrigger.cs*** script to the goal trigger. Drag the goal trigger into the ***GoalTrigger*** field of the goal.
- Create 4 points around the goal mouth. These points mark the area/frustum the goal keeper will try to defend when a shot is made. These points are relative to the forward direction of the goal. Drag each point to its relative field in the goal.
- Save the goal keeper as a prefab.

GOAL KEEPER

- Create an empty game object in the scene and name it ***GoalKeeper***
- Attach the ***GoalKeeper.cs*** script to the ***GoalKeeper*** game object. The required components will be added to the ***GoalKeeper***.
- Attach the ***GoalKeeper*** model to the ***GoalKeeper*** and just with the ball make sure the model stands on the ***GoalKeeper***. Add the ***GoalKeeperAnimationEvents.cs*** to the ***GoalKeeper*** model. Drag the ***GoalKeeper*** in the ***GoalKeeperField*** of the ***GoalKeeperAnimationEvents***.cs script. Initialize the ***GoalKeeperModel's*** animator's controller with the ***AnimationController*** saved in the Animators folder. Drag the ***GoalKeeperModel*** into the animator field and the ***GoalKeeperModelRoot*** of the ***GoalKeeper***.
- Adjust the Capsule Collider to match your model.
- Set the ***GoalKeeper*** rigid body as follows:
  - UseGravity = true
  - IsKinematic = true
  - Interpolate = Interpolate
  - Collision Detection = Continuous Dynamic
  - Constraints = Freeze Position Y; Freeze Rotation X, Z.
- Adjust the GoalKeeper's attributes:
  - Dive Speed = the max speed that the player can dive with.
  - Goal Keeping = how good the goal keeper is at protecting the goal.
  - Jump Distance = more like the dive distance, this is the maximum distance the player can cover when he dives.
  - Jump Height = the maximum height the player can jump (displacement).
  - Reach = the distance from the player's position that he can reach if he stretches his hand.
  - Tend Goal Distance = the distance from goal the player should protect the goal.
  - Tend Goal Speed = the speed of the player when tending the goal.

SETTING UP THE DEMO SCENE

INTRODUCTION

In this chapter you will learn how to set-up the demo scene

SETTING UP THE STADIUM

- Drag the stadium model in to the scene.
- Add a cube in the scene. You can name it to pitch. Scale it so that it covers the entire pitch's floor. Position it so that it's flush with the stadium's pitch and disable the mesh renderer.
- Drag the goal prefab into the scene and place it at your preferred position.
- Drag the ball into the scene and place it at your preferred start point.
- Drag the ***GoalKeeper*** into the scene and place him at the goal. Drag the Ball and the Goal into their relative fields in the ***GoalKeeper***.
- Create an empty game object in the scene and name it ***GameManager***. Add the ***GameManager.cs*** script to it. Drag the various entities in the scene into their respective fields in the ***GameManager*** game object. Create the Canvas and add the text component to it. Drag this text component into the score text field of the ***GameManager***. You can adjust the Game Manager:
  - Ball Dribble Force: the force used to kick the ball around
  - Ball Shoot Force: the force used to shoot the ball at goal
- Create cube behind the goal and stretch it cover a large area. Set the layer of this cube to be ***RaycastReceiver***.
- Add the camera to the scene and position it to your liking. Untick the Goal Trigger and the ***RaycastReceiver*** in the culling musk.

PLAYING SUPER GOALIE BASIC

INTRODUCTION

In this chapter you will learn how to play Super Goalie Basic

PLAYING SUPER GOALIE BASIC

- Use W, A, S, D to move the ball around
- Click on position to shoot the ball towards it
- Wait a little bit for the next round

INTEGRATING WITH SUPER GOALIE BASIC

INTRODUCTION

I this chapter you will learn how to integrate Super Goalie Basic with an existing package

HOW SUPER GOALIE WORKS

As the name suggests, Super Goalie Basic is a basic implementation of a football goal keeper. The keeper is able to protect the goal and tries to make it difficult to score a goal by moving around the goal area and find a position that makes it difficult to score a goal and to intercept the ball when a shot is made.

The artificial intelligence for this package is modelled as state machine. The goal keeper has numerous states and these are:

- ***IdleMainState***: this the initial state. In this state the keeper simply does nothing
- ***TendGoalMainState***: In this state the keeper tries to find a position in front of the goal that makes it difficult to score a goal. It also listens to the ***OnBallLaunced*** event.
- ***InterceptShotMainState***: In this state the keeper tries to intercept the shot
- ***PunchBallMainState***: In this state the keeper punches the ball away from the goal mouth
- ***IgnoreShotMainState***: In this state the keeper just ignores the short, like nothing happened

The goal keeper requires the reference to the ball and the goal for it to work. You can programmatically initialize these two variables in your code or drag and drop the goal and the ball prefab in the scene into their specific fields in the goal keeper. The goal keeper needs to be registered to the ***OnBallLaunched*** event of the ball. Check the ***GameManager.cs*** on line number 45. This helps the keeper to listen to the shot events of the ball and react to them.

You will need to launch the ball using the Launch function of the ball. Check the ***GameManager.cs*** script on line number 94 to 97.

INTEGRATION INSTRUCTIONS

- Import Super Goalie Basic into your project
- Set up your scene as preferred. It depends whether you want to dynamically create your scene or have prefabs already positioned into scene as in the example.
- Initialize the goal and ball prefabs of the goal keeper. You can write logic to do this or you can follow the instructions in this manual if you are using a similar setup.
- Make sure the goal keeper's ***Instance_OnBallLaunched*** function has been registered to the ball ***OnBallLaunch*** event.
- At this stage Super Goalie Basic is ready to work with your project.

YOU MIGHT ALSO LIKE

- Intelligent Selector
- Robust FSM
- Soccer AI