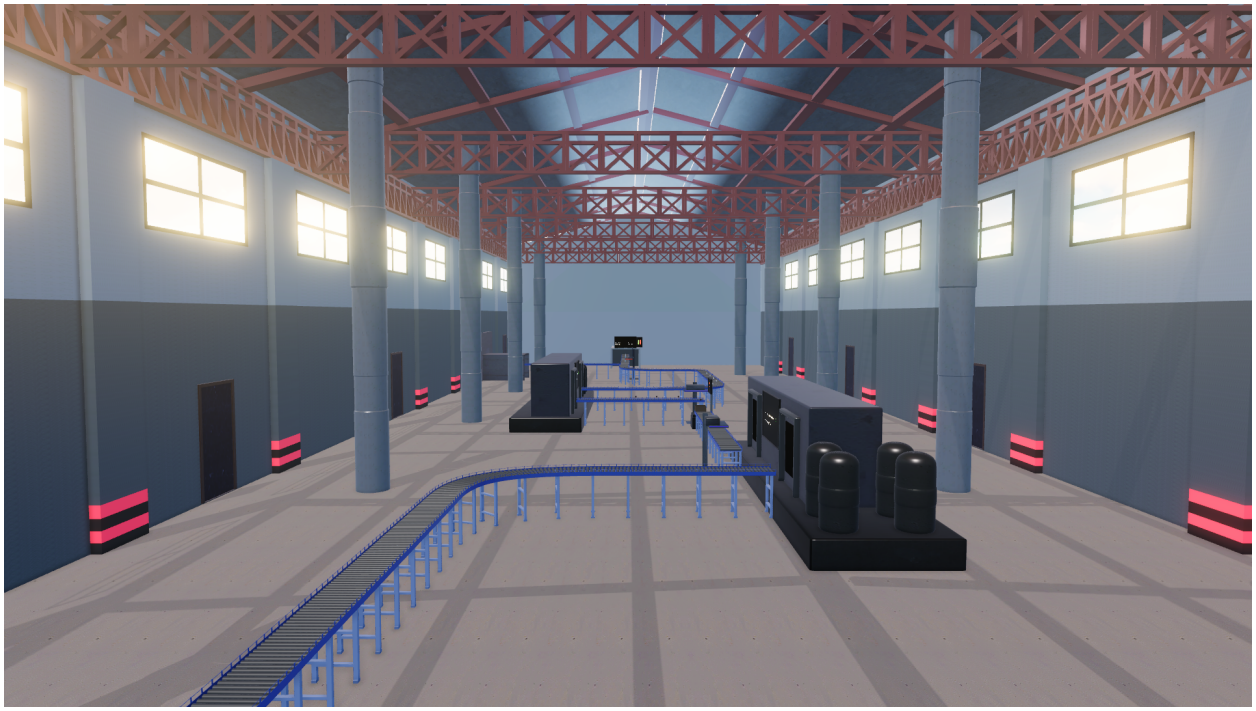

Factory Simulator

Olusola Olaoye

On the unity asset store

Introduction

Factory simulator is a package that contains tools to help you create realistic simulations for production lines in factories. This package could be used as a building block for both games and serious simulations.

How to get started

To get started, open the demo scene that comes with this project and try to play around with it to see what's going on. This package uses the built-in render pipeline by default.

URP pipeline

If you import this project into URP, then you need to

- Click on the window menu, under the rendering option, click on render pipeline converter.
- Click on the dropdown and select the "built in to URP" option.
- Check every setting below (Rendering settings, material upgrade, etc).
- Click on initialize converters.
- Then click on convert assets.
- After that's done, search for a script called "ComputerScreens" in the project files and open it.
- Uncomment the line that contains the code
  screen_material = new Material(Shader.Find("Universal Render Pipeline/Lit"));
- Then comment out the line that has the standard shader.

HDRP pipeline

If you import this project into HDRP, then you need to

- Click on the window menu, under rendering, click on HDRP wizard.
- Scroll down and click on “convert all built in materials to HDRP”.
- Then search for a script called “ComputerScreens” in the project files and open it.
- Uncomment the line that contains
  screen_material = new Material(Shader.Find("HDRP/Lit"));
- Then comment out the line that has the standard shader.

Using this package

Open up the demo scene that comes with this project and play around for a bit. If you wish to set up a factory scene, then open up a new scene.

If the new scene has a camera, you can just add the **CameraController** script as a component to the camera. Another option will be to right click on the hierarchy panel, then go to the **factory simulator** option and click on **sim_camera**. There are 14 objects you can create with the factory simulator menu. Each object has a specific role to play in helping you create your production line simulation. I will go through each one.

Sim Factory

This object has no script. It is just the 3d model of a factory building. Feel free to use your own model if you have one.

Sim Accept Reject Machine

This creates an object called "**Accept Reject machine**". The object comes with a box collider component as well as a "**Accept reject Machine**" component. The component has two vectors. One vector is the accept vector which is the direction of the force that will be applied when a non-faulty item collides with this object. The other vector is a reject vector which is the direction of the force that will be applied when a faulty item collides with this machine.

Sim Conveyor belt

This object has an array of transforms called paths. These transforms are child objects of the conveyor belt object and will dictate the direction for items in the conveyor belt. It holds reference to the conveyor belt collider which lets the conveyor belt know how many items are on it. There is also a **constant speed** boolean variable which allows you to decide if items will move at a constant speed or a fluctuating speed (determined by how spaced apart the path transform child objects are). Finally there is a movement speed variable you can adjust for the moving speed of the items.

Sim in out machine

Has access to the input collider which is a type "**in-out machine collider**". It allows us to store every item that collides with it in a queue data structure. The **output transform** allows us to dictate where the items will be spawned out from when they collide with the input collider object. Processing speed is basically the time it takes (in seconds) for items to be spawned in the output location after colliding with the input. **Output vector** is the force applied to the object immediately after it is placed in the output location.

Sim Item maker

This is the object that is responsible for spawning (producing) the items. Items are spawned in batches. You can choose how many items are spawned in a batch (**batch output**), how often those items are spawned (**output frequency**), and also the delays between batches (**batch delay**). **Quality rate** determines the chances of the item being non-faulty. A

quality rate of 1 will ensure all items are not-faulty and will not be rejected by an **accept reject machine**. **Spawn location** is the location at which the items will be spawned. **Spawn force** is the force applied to the object when spawned. Then there are **3D buttons** which allow you to increase and decrease different values when the program is running. The 3D buttons change certain values when they are clicked on.

Sim output collection

This class is simple. We simply increment the variable that stores the number of **items collected**. Items collected are basically items that collide with this object.

Sim package handler

This object simply spawns a new **container** object at the **input location** and when that container object is full, it sets the container's position to the **output location** and then spawns a new container.

Sim pusher machine

This object pushes any item object that collides with it in the direction of its **push vector**.

Sim reject container

This object simply destroys any object that enters its trigger collider. These objects are typically items that have been rejected by the Accept reject machine (faulty items).

Sim Time manager screen

Simply shows the **time elapsed**, **time started** and the **current time** all in date time format.

Sim camera

This is a camera object with a camera controller script attached to it. The script allows you to adjust **movement speed** (for up, down, left, right movement), **turn speed** (for pitch and yaw rotation) and **zoom speed** (for forward and backward movement). It makes your camera function like the scene view camera.

Contact Me

Feel free to contact me on "olusola.i.olaoye@gmail.com" if there are further questions concerning this project.