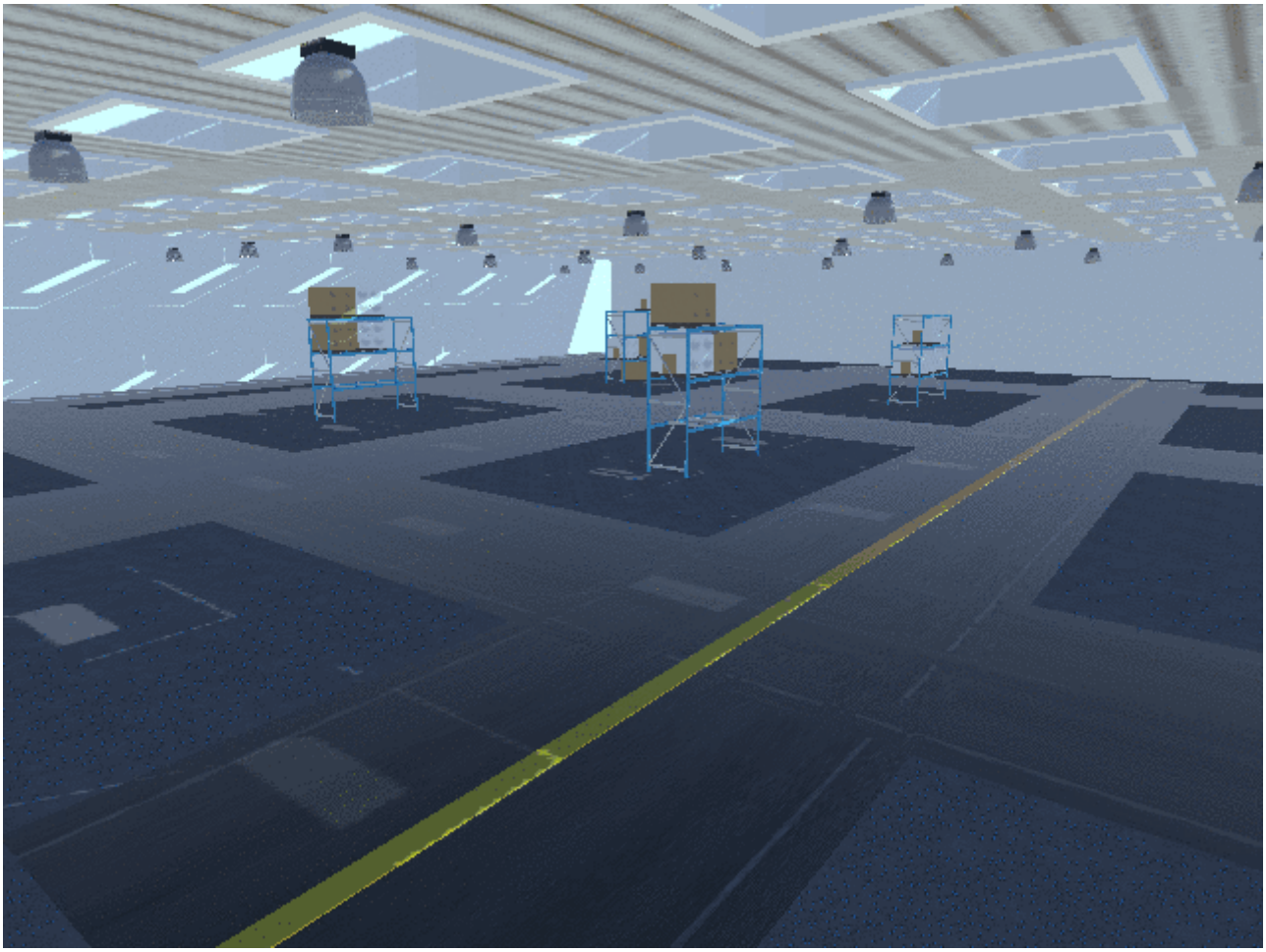
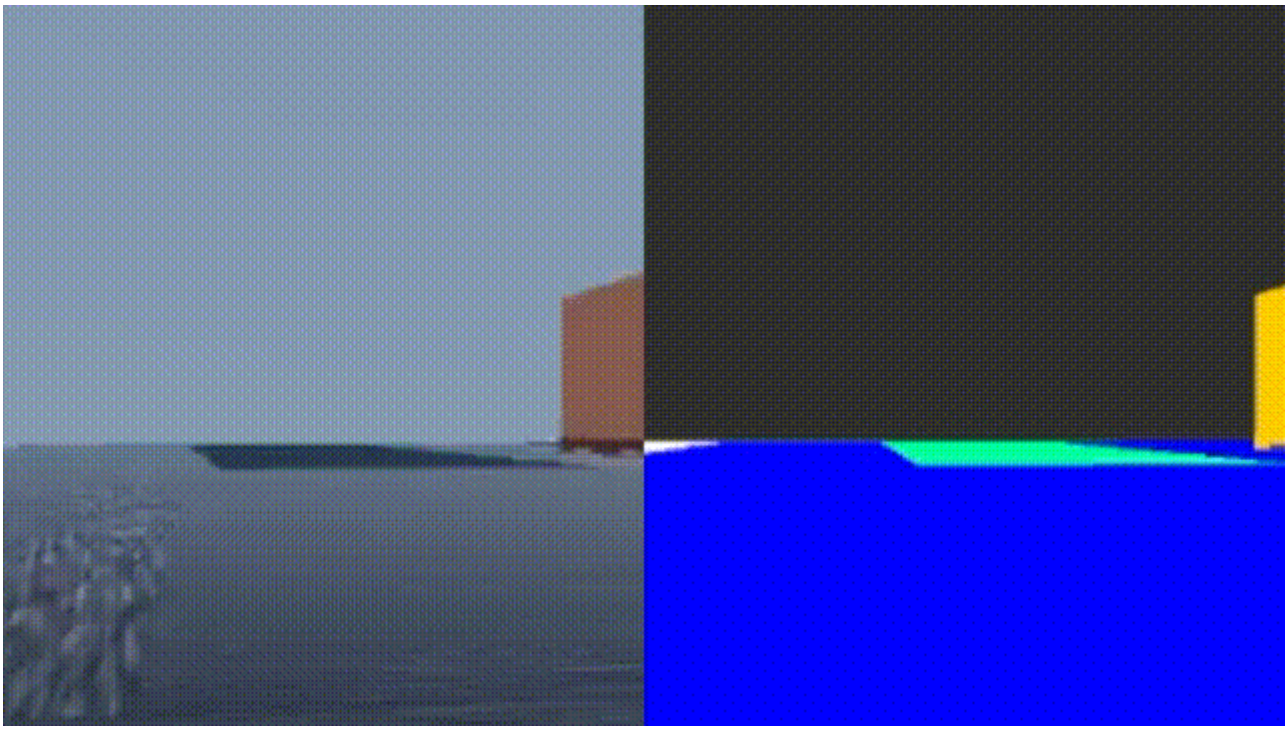
Unity Simulation Sample Project

Warehouse Robot

Rendering Pipeline : Universal Rendering Pipeline

This sample project demonstrates the usage of Unity for environment generation, Perception SDK for RGB images and semantic segmentation, and Unity Simulation for running it at scale to generate a dataset that can be used for training a machine learning model.





About Project

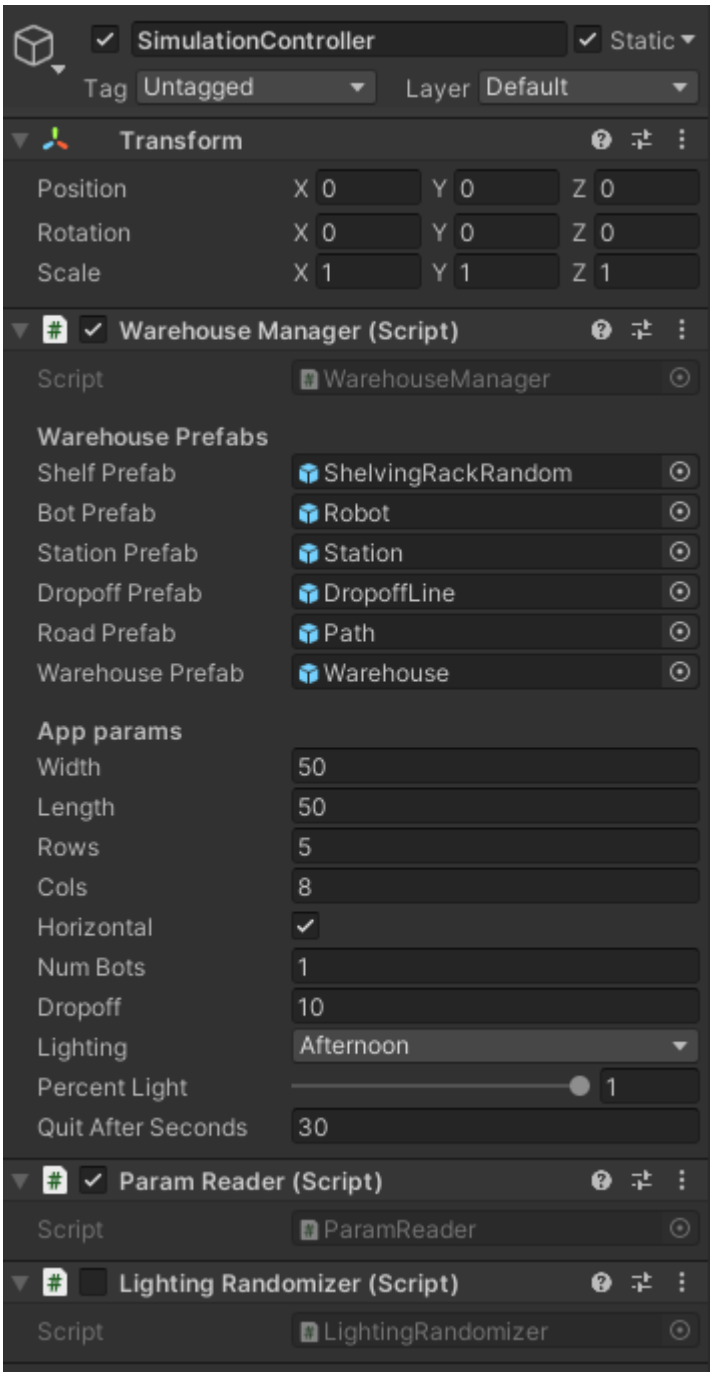
Category : Industrial Robotics

This project is an example of building a synthetic dataset using generated environments. The scene setup begins empty, as the warehouse layout is generated via the input parameters. The robot contains a forward-facing camera, and navigates between pickup and dropoff points throughout the warehouse.

Components

SimulationController

This is a GameObject in the hierarchy that is responsible for driving the simulation. Components on this object read in the appParams (`ParamReader.cs`), instantiate the warehouse setup (`WarehouseManager.cs`), and randomize the lighting (`LightingRandomizer.cs`).



Robot

This object contains the NavMeshAgent and associated controls to enable the bot to navigate to waypoints. The `RobotAgent.cs` component drives the robot behavior.

PerceptionCamera

The PerceptionCamera enables capture of RGB images, semantic segmentation and bounding box annotations for the source camera on which this script is added. In this project, the PerceptionCamera component is added to the robot's camera. The output data folder is printed in the Console window of the Unity Editor, e.g. `DC[V]: Output path set to : /Users/<user>/Library/Application Support/DefaultCompany/Warehouse-Robot/<ID>/Logs`.

CustomReporter

The CustomReporter uses the Perception SDK DatasetCapture capabilities to track additional metrics during each run, including the bot's current rotation, position, and current pickup/dropoff target and next navigation destination. Some sample data outputs are listed below.

Custom metric definitions allow for data to be logged at each time step, such as the NavMeshAgent next path coordinate.


```
"metric_definitions": [
{
  "id": "4dec65b1-3376-47e1-bf9e-deefb0926cc1",
  "name": "Next corner",
  "description": "The next node in the NavMeshAgent path"
},
```



```
{
  "capture_id": null,
  "annotation_id": null,
  "sequence_id": "8f769018-5eb3-4abb-9c0c-8b625ba41bb2",
  "step": 0,
  "metric_definition": "4dec65b1-3376-47e1-bf9e-deefb0926cc1",
  "values": [{ "x": -13.66667, "y": 0.04333322, "z": 2.83333 }]
},
```


Data of the current location of the perception sensor--in this project, the robot--is logged by default.

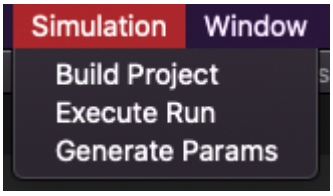

```
"ego": {
    "ego_id": "1927c013-ebdd-476c-b0c0-acc44cc9b7cd",
    "translation": [
      -13.0009689,
      0.143333226,
      -24.9602184
    ],
    "rotation": [
      3.31298812E-12,
      -0.00231897179,
      -3.50368E-09,
      0.9999973
    ],
```


Further documentation on the Perception SDK can be found here.

Run on USim

In order to run your simulation on USim, you will have to sign up for the service by visiting our website.

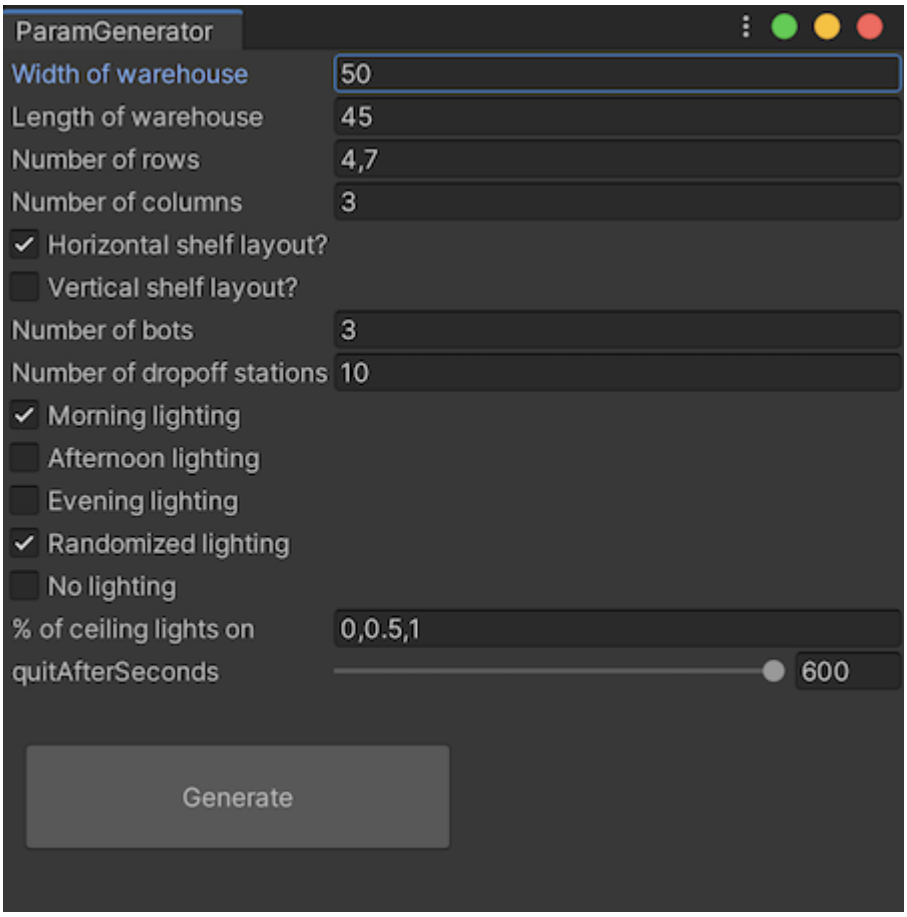
You can run your simulation on USim using the USim CLI by following instructions here. Alternatively, you can use the Simulation Client package, which exposes C# APIs to automate USim CLI workflow in Unity Editor to improve iteration time. This project demonstrates usage of Client package APIs by creating a menu item, which provides options to Build Project and Execute on USim. Build Project will perform a Linux Standalone build and zip it up for upload. Execute menu item will create a run definition to run the simulation with the build generated using vCPU6 sysparam and schedule it on USim. After the run is scheduled, it will log the execution-id to the console. Make sure you are logged in to Unity Services and your project is linked to a cloudprojectId.



Parameterization

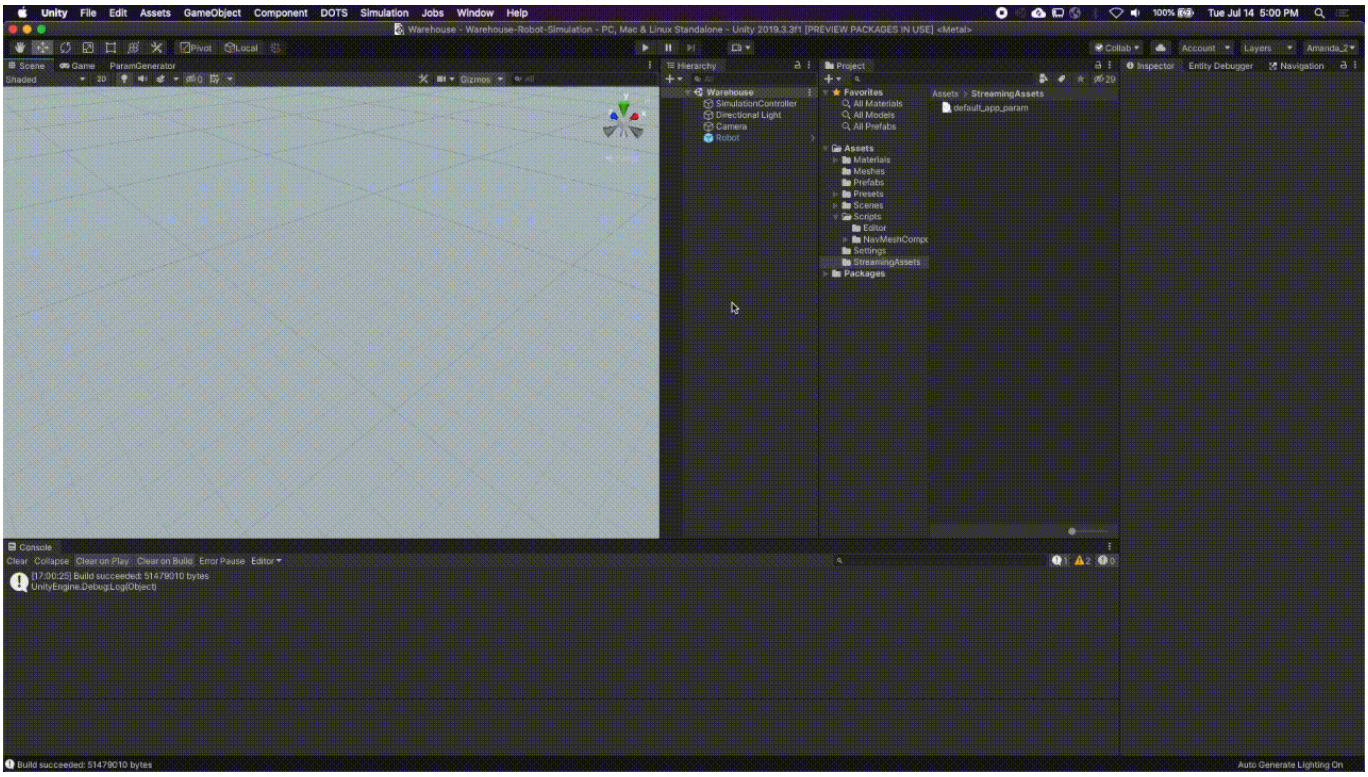
Parameterization in this project is what enables differing warehouse layouts. This allows for variety in the generated data. Here are the options for paramaterization:

- Width and length of the warehouse
- Number of rows, columns, and orientation of shelves
- Number of bots instantiated
- Number of dropoff stations
- Lighting type (Morning/Afternoon/Evening, Randomized, None)
- Percent of ceiling lights turned on



You can set different appParams with these options and run the desired number of instances against each appParam. The simulation config is parsed on `Awake` and applied to the simulation configuration.

In `Simulation->Generate Params`, all combinations of input app params can be generated as JSON files to Assets/StreamingAssets.



How to extend this project

This project can be extended to simulate robot navigation in differing environments. The robot's navigation system can be replaced with a custom model, trained on the synthetic dataset generated in the USim runs.

Limitations and known issues

Limitations

- The number of bots generated is limited to the warehouse `width / 2`.
- The number of columns and rows of shelves are limited by the length and width of the warehouse.
- The number of visible ceiling lights is limited by the graphic settings of the Unity Editor.
- Performance may decrease when the warehouse dimensions exceed `200x200`.

FAQs

1. The `Execute` menu option doesn't work for me

```
This might typically happen if you are not logged in or linked your
project. You can do this by opening the Service window (Cmd + 0 or Ctrl +
0) and clicking on SignIn and link your project.
```

2. I am logged in and my project is linked, but I am still not able to run on USim

```
This might happen if you have not signed up for the Unity Simulation
Service.
```

3. What are all these `The type or namespace name '...' does not exist` errors?

```
This is a sign that the package dependencies have not been installed
correctly. When importing the project for the first time, a popup
"Warning: This Unity Package has Package Manager dependencies" will show.
Click "Install/Upgrade."

Ensure that the Package manifest(located in Warehouse-Robot-
Simulation/Packages/manifest.json) exists, and contains at least the
following:

"com.unity.perception": "0.2.0-preview.1",
"com.unity.render-pipelines.universal": "7.1.8",
"com.unity.simulation.client": "0.0.10-preview.9",
"com.unity.modules.ai": "1.0.0"
```

4. Nothing is happening when I click Play

```
Make sure you have opened the Warehouse Robot/Scenes/Warehouse.unity scene
in the Unity Editor before you enter Play Mode.
```

5. I'm getting some weird errors!

```
This project works best when imported into a new, empty Unity project. If
the problem persists, check out one of the links below.
```

Try visiting our Unity Simulation forum, or email us at simulation-help@unity3d.com for more help.