Assignment 5 Report
Joshua Chung, David Gwizdala, Alexander Tejada, Geordon Worley

Abstract: This report will discuss our REST APIs and external testing of the assignment on providing a storage solution for pathfinding as well as a robot simulation.

Informational website: https://ldevr2t1.github.io/

Pathfinding:

- **Documentation**
- **Repository**

Robot Simulation:

- **Documentation**
- **Repository**

REST API - Pathfinding Storage

Our subgroup created a YAML configuration file for Swagger to generate our server and web documentation. We documented the consistency model, usage, and JSON structures via the swagger documentation which can be seen here. We run on a AWS EC2 box and the v1 API is hosted at http://ec2-52-41-229-1.us-west-2.compute.amazonaws.com:8080/v1/. This simple storage service was sufficient and therefore we did not produce another protocol version. It works by giving clients generated unique IDs to store and retrieve JSON formatted data from and to. We use MongoDB under the hood to store this data in a consistent format so that the server code itself is fully stateless.

We initially attempted to use TOX to test the server, since swagger generated TOX test code, however after several hours we were unable to run tests using this framework, so we ended up writing a standalone client to test a remotely running server with a series of tests which attempt to break it. This client may be found here. It only requires Python 3 and requests to run. requests can be installed via pip: `pip3 install requests`

A video was captured of this test client running and showing the client's code briefly and can be found here.

REST API - Robot Simulation

We used Swagger to create our skeleton for our server. We then used the built in functionality to convert the skeleton to Python code that integrates a Flask server. Our server stores problem information of any object type, allowing the team responsible for the robot solution to build and update their objects in any way they'd like.

External Testing

The external testing, located in /server_testing at https://github.com/ldevr2t1/RoboSim, contains the following:

- test_problems.py (Unit test for Problems)

A video showing test case completion can be found at https://github.com/ldevr2t1/ldevr2t1.github.io via the **/Submission_Videos/AS5_RoboSim.mp4** path.

Overview

We use the Nose testing suite to determine if a proper status response has been returned from our database. We run through all possible api calls, pre-generated by swagger in their own respective test folder, and compare them to the required output. If an unexpected status code arises, a failure code is generated for review. We test for appropriate and inappropriate requests and responses, including:

- Getting all problems
- Posting a new problem
- Getting a problem via a specific id
- Attempting to get a nonexistent id
- Updating a problem via a specific id
- Attempting to update a nonexistent id
- Deleting a problem via a specific id
- Attempting to delete a nonexistent id

How to Run

The nose testing suite and requests package must be installed on your machine for this test to function. If you do not have these packages please install using:

```
pip install nose
pip install requests
```

To run the tests, navigate to the server_testing folder and type the following command into your console:

```
nosetests
```

The rest is done automatically.