LiDAR-based navigation in MuJoCo

This repo contains code to simulate – **using MuJoCo** instead of ROS or Gazebo – a simple differential drive robot navigating the BARN challenge worlds (download zip file) using a LiDAR sensor for feedback.

Simulating a world

Run

```
python3 sim_world.py <world_number>
```

Optional arguments:

- x0: initial x position in meters
- y0: initial y position in meters
- t0: initial θ angle in degrees
- nsteps: number of steps to run sim
- xml: which environment to simulate
- prefix: prefix of sim file name
- suffix: suffix of sim file name

Run this command for defaults:

```
python3 sim_world.py --help
```

Due to the size of the arena, a common choice is:

```
python3 sim_world.py <world_number> --y0=7.0
```

Currently, the controller used to map LiDAR into body velocities and then wheel speeds are hard-coded. Also see controllers.py. You can modify the code to test your own controller.

The differential drive robot with LiDAR is defined in differentialdrive.xml. It loads in currentxml.xml, onto which the correct world is copied during execution.

Plots of a simulation

The sim_world.py file generates three files with names containing the <world_number>. These can be processed to generate images using

```
python3 generate_plot.py <world_number>
```

You may need to give the same options that were given to sim_world.py if not using the default ones.

Visualizing a simulation

A separate script is used to visualize the simulation. Run:

```
mjpython visualize_sim.py <world_number>
```

Unlike sim_world.py, the control frequency is the same as the simulation rate. Also, no simulation data are stored.

Odometry

The sequence of scans stored after using sim_world.py can be used to estimate the path of the robot through scan matching of consecutive scans. The odometry script is not currently not optimized. Also, no map is generated.

The sim_world.py file generates three files with names containing the <world_number>. These can be processed to generate images using:

`python3 odometry.py <world_number>`

You may need to give the same options that were given to `sim_world.py` if not using the default ones.

Helper scripts

The short scripts

- `run_sim_all_worlds.py`
- `generate_all.py`

enable evaluation of a controller (currently hard-coded) in all worlds, or some desired subset (currently hard-coded).