766 Midterm Report

John Balis, Yisen Wang, Milica Cvetkovic

March 2021

1 Introduction

We are interested in determining how augmenting input features using values derived from SLAM affects the performance of standard reinforcement learning approaches in simulated 3d environments. To perform this evaluation, we will be using gym-miniworld as our primary simulation environment. This is a minimalist 3d reinforcement learning environment with navigation based reward.
Our current progress has strictly followed the milestones proposed in our proposal. We will demonstrate them in two subtasks, DQN and SLAM.

2 DQN test

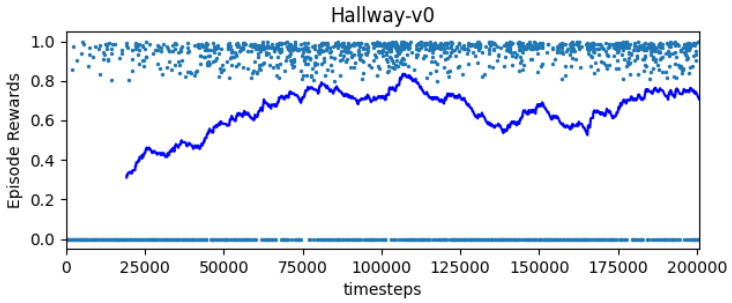

Figure 1: Average episode reward over 200000 timesteps

In order to evaluate how SLAM affects the performance of common reinforcement learning algorithms in 3D environments, we need to establish a baseline performance of reinforcement learning without using SLAM for feature extraction. To provide this baseline, we showcase how average reward changes across many episodes while training a convolutional deep Q network on a hallway navigation task. We use the default convolutional neural network implementation for DQN provided by stable-baselines. This architecture is similar to, and likely inspired by, the architecture used in [3], and consists of a basic convolutional neural network with rectified linear activation functions followed by at least one fully connected layer.

3 SLAM

3.1 SLAM theory

We primarily went through two papers, ”LSD-SLAM”[1] and ”Orb-SLAM2”[2], which are representatives of direct SLAM and indirect SLAM respectively, to catch the core ideas of SLAM. We also read a classic monocular SLAM paper ”MonoSLAM”[4] carefully to dive into the details about how each part works, such as the depth estimation, Kalman Filter Update, feature initialization and map management. Such material gave us a great opportunity to reflect on what we have learned in class and how they come into play in real-world applications.

3.2 Orb-SLAM2 in miniworld

As we discussed in the proposal, gym-miniworld is our simulation environment for testing Q-learning with the help of SLAM. We have incorporated real-time SLAM library implemented in [3] into gym-miniworld by feeding 600x800 observation image into its framework at each step. During the first few steps, SLAM is initializing the map and only uncertainty of the feature depth are shown on the screen. After SLAM took enough data to restore the position and viewpoint of features, corners,bricks and other features are labeled correctly on screen, and the algorithm starts to output localization and mapping results. Fig 2 demonstrate the moving trajectory of SLAM inside a hallway with box.

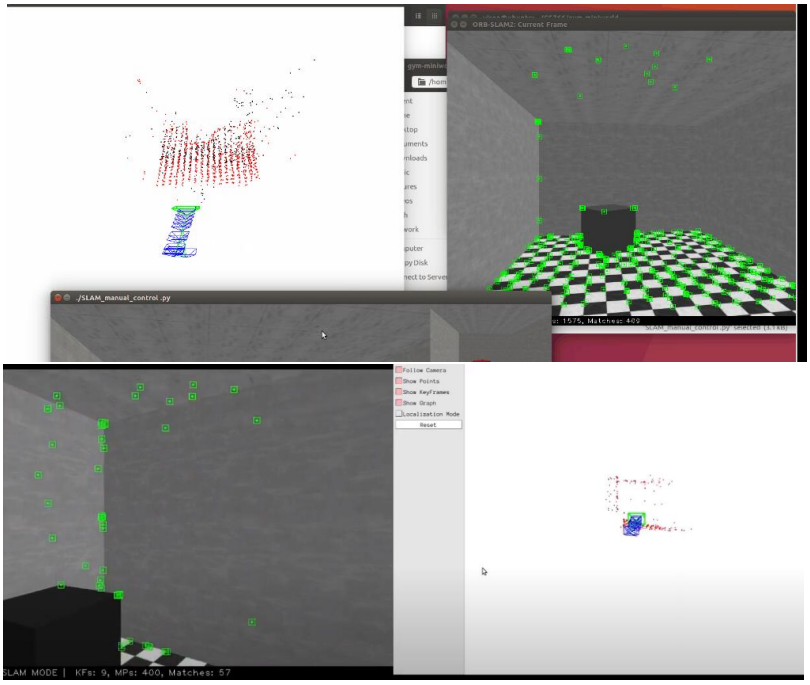

Figure 2: SLAM in miniworld

Ongoing tasks include parameter tuning and testing in a bigger maze. In our testing, we noticed that it

takes longer time to initialize SLAM features in miniworld than in the KITTI benchmark test. The reason could be the discrepancy in the richness of texture between simulated and real-world environments: when in simple simulated environments, the environmental geometry may be patterned with repetitive and in some cases low-contrast textures, which may make it more difficult for the SLAM algorithm to locate and track features.
It's worth mentioning that everyone in this group encountered great difficulties in setting up and installing the system. In fact, we spend average 8+ hours in installing Orb-SLAM2 and python bindings on Ubuntu 16.04, 18.04, 20.04, Mac OS. It turns out only Ubuntu 16.04 and 18.04 works well. A lot of bugs need to be addressed. We actually had to modify the source-code of Orb-SLAM2 to get it to work on Ubuntu-18.04

4 Remaining Roadmap

Following our original project timeline, our remaining goals are to design an input encoding for SLAM features into DQN, to contrast the performance of DQN on miniworld to DQN with SLAM on miniworld, and to complete the project writeup and presentation. We are planning to complete these goals by their original deadlines listed in the project proposal. Additionally, we remain interested in using the Minecraft-inspired reinforcement learning environments from the mine-rl project and applying the findings from our project to solve Minecraft.

5 Website

https://cmilica.github.io/cs766project/

References

[1] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM," Computer Vision – ECCV 2014 Lecture Notes in Computer Science, pp. 834–849, 2014.

[2] Mur-Artal, Raul, and Juan D. Tardos. "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras." IEEE Transactions on Robotics, vol. 33, no. 5, 2017, pp. 1255–62, doi:10.1109/TRO.2017.2705103.

[3] Mnih, Volodymyr, et al. "Human-Level Control through Deep Reinforcement Learning." Nature, vol. 518, no. 7540, Feb. 2015, pp. 529–33, doi:10.1038/nature14236.

[4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 6, pp. 1052–1067, 2007.

[5] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel and Juan D. Tardós, ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM, Under review.