CSE 571 - Robotics
Project 2

1 Description

The project 2 should involve implementing methods that address the mobile robotic problems such as localization, mapping, SLAM, and path planning. The methods will potentially make use of neural networks in one way or the other for object detection, pose estimation and generating low-level control policies. We have provided you a list of project ideas as suggestions that can inspire your group's project proposal. We expect it to be a project with the duckiebot unless there is a strong proposal to pursue otherwise. In other words, we welcome strong open-ended project proposals that are related to robotics. All of these proposals have to be approved by the course staff.

Note: This project will be completed in teams of three.

2 Deliverables

The proposal, mid-progress report, and the final report should be submitted as PDF files on gradescope. The code should be submitted through the private repository shared with course staffs.

Proposal [1 page] The proposal should include:

- The team members (full names and uwnetids) working on the project and each team member's expertise/skill sets.
- A summary of the proposed project or the final robot demonstration we expect to see.
- Specific details including, inputs and outputs, modules in the proposed pipeline.
- Any assumptions that will be made in the implementation.
- Evaluation metrics to evaluate your implementation.
- Optional tasks and contingency plan.

- Planned work load among the team members.
- A rough timeline with a list of milestones.

The instructor/TAs will provide feedback to help guide you.

Mid-Progress Report [1-2 pages] A progress report of successes and unforeseen problems. If the timeline/project outcome needs to be updated, please note this in the mid-progress report.

Final Report [3-5 pages] The final report will summarize the project. Describe the task, problem of interest, and the (environment) setup, approaches, evaluation results in the form of figures-tables-plots, qualitative results on a duckiebot with videos of it (youtube links in *unlisted* mode). Please include any references if you are building off of them (references are excluded from the page count). In addition to the final report (that will have video links to the experimental results), teams must submit a 8-minute presentation video describing the project/results. Refer to conference research videos (example - `https://tinyurl.com/jcxz4t25`) that follow motivation-objective-approach-results format.

Code Submission Similar to project 1, create a private github repository and share it with the course staff.

3 Timeline

- Proposal: due on 4/29/21.
- Mid-Progress Report: due on 5/13/21.
- Final Presentation: 6/8/21.
- Final Report and Code: due on 6/10/21.

4 Suggested Project Ideas

The following projects are provided as suggestions. You can mix-and-match from these ideas to create a final project proposal. Some of these project ideas will probably require more work than is possible in the time allocated for this course. Your project proposal should detail the modules you plan to implement and the modules you might borrow from existing works or project 1. When you are writing up your project proposal, try to think about what you will be able to accomplish realistically. It will also be beneficial to include what the team proposes to accomplish along with contingency plans and optional tasks.

4.1 SLAM related project ideas

In project 1, you will be implementing EKF and PF methods to localize the duckiebot given a map and control inputs via teleoperation. Imagine your duckiebot was placed in a new duckietown it has not been to before. All it can do is drive on a road following lane rules (staying right of the yellow lines). The objective of this project would be that the robot will have to detect signs (landmarks) and build a map of the town while navigating in the town. This will involve localization on a mapped town while simultaneously mapping the environment. While the SLAM method could be of your choice, the project will have to include land following controllers, along with some of the below-mentioned variations/extensions. Some variations/extensions:

- **SLAM without fiducial markers**: Training a detector for road signs along with pose estimation of these signs (and not use the AprilTAGs). Training data can come from an AprilTAG detector. One could potentially evaluate their SLAM implementation with and without AprilTAGs.
- **SLAM with duckies and road signs as landmarks**: The landmarks can also include duckies. In addition to using a duckie detector, this will include data association problems, i.e., distinguishing between already seen landmarks (duckie) and the newly seen landmark. At the end of the mapping, you should give the total number of unique duckie citizens in the town.
- **SLAM with road signs as landmarks and duckies as moving citizens**: Imagine a pedestrian crossing the road or another car driving in front of you. In these cases, you would not want to use features on these objects because they move. One way to account for this is to train a network that masks out regions of an image that you know to be problematic. You could train a masking model on the duckies and fold it into a SLAM pipeline. You would show the effectiveness of this method by randomly repositioning the ducks and demonstrating that your localization performance stays consistent.

4.2 Visual navigation related project ideas

The duckiebot is equipped with a camera and can potentially use the image information to navigate around a duckietown (map). Refer to `https://homes.cs.washington.edu/~xiangyun/topological_nav`.

- **Global localization in a map**: Imagine that the robot has driven around the map many times before. It has data of images collected during these prior experiences. Now given an image from the robot's point of view from a random location in the map, the robot should be able to localize this location in the map. This will involve creating an offline trained model of the map where, given a query image, the model can predict the (x, y, theta) location in which this image was taken from. You have to train a deep-learning model by collecting images in the environment. The robot should be able to navigate from its current location to this query location in the map.
- **Local controller in a map**: Imagine that the robot has driven around the map and stored a subset of the images it has observed on its route. The path taken by the robot in the town can be represented as a graph with nodes and edges, essentially images connected to their neighbors. The robot should be able to retrace its route by simply trying to reach one image after the other. The goal is to learn a controller that uses the current image in order to determine how it can reach a target image that is not too far away.

- The project could be a combination of both these aspects, where the robot can localize where the query image is taken from the map and then decide the nodes it has to connect toward this goal while using the local controller.

4.3 Multi-robot control behaviors

Imagine multiple duckiebots autonomously navigating in the duckietown environment. These robots should navigate from one location in the map to another while following traffic rules while avoiding any collisions. At least one robot should be capable of going from point A to point B in the map, while the other is being teleoperated or having a fixed behavior. Assuming lane following is an essential behavior for this robot, some (if not all) behavioral scenarios to be included are:

- **Traffic sign behaviors**: For example, while moving in a known environment, we as humans will stop when we see a stop sign. The robot will mimic the behavior of stopping at the stop sign while it is moving from point A to point B. In the experiments, this stop sign should be placed randomly in addition to the landmarks that are part of the map.
- **Do not kill the duckies**: Duckie town will have duckie citizens who are not part of the map but will be seen in the town. Sometimes they might even appear on the road. The robot should have reasonable behaviors when it encounters the duckies.
- **Tail/mimic the robot in the front**: When driving in the real world, we probably have encountered moving vehicles in front of us. In these scenarios our behaviors might include a) tailing the vehicle as long as it is in our route, b) mimic their motion behaviors as they might be avoiding a stopped vehicle or pedestrian, c) if this vehicle has a blinker on, then we overtake them by moving onto the left lane.

4.4 Note: Duckietown AI Driving Olympics (AIDO) as a reference

Duckietown community has been operating a set of challenges on lane following tasks in both simulation and real duckiebots. The goal of the challenges is to task a robot to follow a lane with or without obstacles from color images. Baseline algorithms, reinforcement learning methods, dataset aggregation, behavior cloning, and residual policy learning are provided as examples. You can take a look at these tasks, learning-based approaches, or sim-to-real techniques in AIDO to get inspiration for your projects.

For instance, you can train an end-to-end training model to make your duckiebots follow the lanes and navigate in a few customly built duckietown maps. This can also be two phased networks, one that detects the lane first and the other that takes the detection results to output motor controls. The data collection from robots is slow, expensive, and inefficient; training a model from simulated images and simulated actions can be a good starting point. It can be used as a sanity check of your model and the AIDO leaderboard can make this process easier. After getting reasonably good performance in the simulator, you may move on to real world cases considering differences between images from the simulator and the real robot environment as well as the differences in dynamics. For reducing the domain gap, the simulator supports domain randomization and you probably need that too. The next step can be collecting a real world dataset and fine-tuning the model with it. Still, the robot

can suffer from drifting and deviate from the center of the lane; you can either do dataset aggregation, image augmentation, or your own idea to alleviate the problem.

In addition to a basic lane following, you can add some duckie pedestrians and try to make your duckiebot avoid duckie pedestrians. With the object detection dataset, you can train a duckie pedestrian detection model. Another interesting direction is to estimate actions from the recorded driving videos. It is rather an open-ended topic and the related papers are included in the resources section.

Resources

Duckietown documentations and web pages that can be useful for your project:

- Duckietown map assembly instructions `https://docs.duckietown.org/daffy/opmanual_duckietown/out/dt_ops_tiles.html`
- Duckietown AI Driving Olympics `https://www.duckietown.org/research/ai-driving-olympics`
- Duckietown exercises `https://github.com/duckietown/dt-exercises`
- Research using Duckietown `https://www.duckietown.org/research/guide-for-researchers`
- University of Montreal course page `https://liampaull.ca/ift6757/`
- UMass Lowel course page `http://sites.uml.edu/paul-robinette/teaching/eece-5560-spring-2021/`
- Jetson Nano deeplearning documentation `https://developer.nvidia.com/embedded/learn/jetson-ai-certification-programs#course_outline`

Popular deep-learning frameworks that you can use for training models:

- PyTorch `https://pytorch.org`
- Tensorflow `https://www.tensorflow.org`
- MXNet `https://mxnet.apache.org`

Academic conferences where you can find related papers:

- RSS `https://roboticsconference.org/`
- ICRA `https://www.ieee-ras.org/conferences-workshops/fully-sponsored/icra`
- CoRL `http://robot-learning.org`

- IROS https://www.ieee-ras.org/conferences-workshops/financially-co-sponsored/iros
- Computer vision conferences: CVPR, ICCV, ECCV

Companies / organizations that are doing robotics-related research:

- NVIDIA https://www.nvidia.com/en-us/research/robotics/
- OpenAI https://openai.com
- Google Robotics https://research.google/teams/brain/robotics/
- Facebook AI https://ai.facebook.com
- Self-driving related: Uber, Waymo.

Duckietown AI Driving Olympics

- The AI Driving Olympics (AIDO) https://www.duckietown.org/research/ai-driving-olympics.
- AIDO Book (contains some details of tasks) https://docs.duckietown.org/daffy/AIDO/out/.
- Transfer Learning for Autonomous Driving in Duckietown (a project from another course with Duckiebots) https://home.ttic.edu/~cbschaff/duckietown.
- Slide of DAGGER https://www.cc.gatech.edu/~bboots3/ACRL-Spring2019/Lectures/DAgger_slides.pdf.
- DAGGER paper https://www.ri.cmu.edu/pub_files/2011/4/Ross-AISTATS11-NoRegret.pdf.
- Image augmentation for autonomous driving https://arxiv.org/pdf/1604.07316v1.pdf.
- Object detection dataset https://docs.duckietown.org/daffy/AIDO/out/object_detection_dataset.html.
- Object detection video gallery https://bit.ly/3xggAQZ.
- Learning Navigation Subroutines from Egocentric Videos https://ashish-kmr.github.io/subroutines/.