CMSC477: Robotics Perception and Planning
Project 1: Plan Your Way-Out Using Landmarks

Prof. Troi Williams, Anukriti Singh, Khuzema Habib

February 20, 2025
Due: March 13, 2025

1 Introduction

In this assignment, you will learn and implement a robot navigation method for a real robot in a known environment. The environment is a space with landmarks (or fiducials) that can be detected. These landmarks will be AprilTags. You will use the AprilTags to determine the robot's position and orientation (pose) in the world. You will then write a controller that uses the robot's pose to follow a pre-planned path through a maze. The path through the maze will be calculated by your planning algorithm based on either Dijkstra's algorithm or A*.

As demonstrated in Project Zero, if the robot camera is looking at one (or more) April Tags, the position and orientation of the tag in the camera frame can be determined. These six quantities are collectively known as the tag's pose. In lecture, we will go over how to determine the robot's position in the world using the tag's pose and the known position of the tag in the world as given by a map.

- The environment will consist of storage cubes with AprilTag's mounted on them. The storage cubes will be placed on a regular grid according to Figure 1. You will use these landmarks to determine the robot's position.
- You are required to estimate the *shortest* path that the robot will follow. You can use Djikstra (from HW1) or any other planning algorithm that you implement (such as A* or RRT).
- Each AprilTag will have a different tag ID. Figure 1 shows where the tags will be placed and what their ID will be.
- You will be given a starting position and ending position.
- For this project, a map of the environment will be given as shown in Figure 1. The task is to plan a path through that environment and follow that path with the robot. The robot will use regularly placed AprilTags (in known positions as marked on the map) to determine its position in the world.
- The entire map is made from storage cubes of size 26.6 × 26.6 cm.

2 Prerequisites

You will need to use apriltag.py from Project 0 and one of your team member's implementation of Dijkstra's algorithm from Homework 1.

3 Suggestions

You are free to accomplish this project in the manner of your choosing. However, the following are a few suggestions we recommend:

- Use interpolation to smoothly transition between the grid points returned by the path planning algorithm
- Find and use a single AprilTag in the current image to determine the robot's camera pose in the tag's coordinate frame
- Compose the result with the known position of the tag in the world frame to determine the robots pose in the world frame (by looking up the tag's ID in a map)
- Use velocity control to correct for the error between the current robot position and the target position defined by the path

- Use velocity feedforward to ensure the trajectory tracking error goes to zero
- After planning a path, select a target AprilTag to look at every time
- During path execution, keep the target AprilTag in the center of the field of view to help ensure the pose estimate is stable

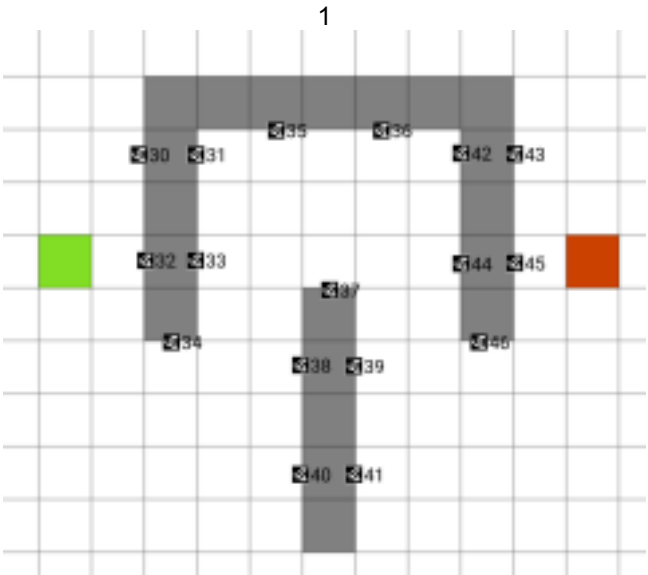

Figure 1: Figure shows the top view (left) and the perspective view (right) of the scene structure. The green block represents the starting position and red block represents the goal position.

4 Extra Credit 1

Solve the same problem – going from start to goal location but without knowing the map. You may want to place AprilTags on all the visible faces of the storage cubes.

5 Extra Credit 2

This is a challenging extra credit, but it should be very rewarding.

Solve the same problem, but use image-based visual servoing to track the trajectory. This is a follow-up to the second bonus from Project 0 based on the paper "Visual servo control. I. Basic approaches".

In image-based visual servoing, the desired trajectory is where the corners of an AprilTag should be in the image at each time. To calculate this trajectory, you must transform the planned path from a sequence of robot positions in the world to a sequence of tag positions in the camera frame. The tag positions in the camera frame can then be used to calculate the desired pixel position of the tag's corner. Finally, the image-based visual servoing algorithm will move the robot to keep a tag's corners at the desired pixel positions.

Indirectly, this will cause the robot to follow the desired path in the world frame. However, it should turn out to be more resilient to noise than doing control based on an estimate of the robot's position in the world.

5 Grading

- Navigating from Start Position to End Position (70 points)
- Report and video (30 points)
- Extra Credit 1 (5 points)
- Extra Credit 2 (10 points)

6 Submission Guidelines

All the files must be compressed into a single .zip file. Along with your code submission and video, your PDF report should contain the following (along with any relevant plots or visualizations):

- Introduction
- Block Diagram
- Link and description of the video (Google Drive or YouTube)
- Methodology
- Results (must include 2D plot showing planned path and path traveled)
- Conclusion
- Code listing

Also submit a video of your demonstration (.AVI/.MP4/.MOV).

7 Collaboration Policy

You can discuss the assignment with any number of people outside your group. But the solution you turn in MUST be your own (as a group). There is only one submission required per group. Please add the group member names at the top of the PDF report. Plagiarism is strictly prohibited. A plagiarism checker will be used to check your submission. Please make sure to cite any references from papers, websites, or any other student's work you might have referred to.