Control of Mobile Robotics | CDA 4621 | Lab 3

Vision-based Navigation & Obstacle Avoidance

Julie Gibson | Jennifer Patterson

# List of Code

* ThreadedWebcam.py
  + Used to implement a threaded webcam.
* UnthreadedWebcam.py
  + Used to implement an unthreaded webcam.
* Calibration.py
  + Used to run a calibration
* bugAlgorithm.py
  + This program implements “bug 0” algorithm to find a specified goal.
* faceGoal.py
  + This program rotates robot around its center until it faces one of the specified goals.
* Kinematics.py
  + This program contains kinematic functions
* Movement.py
  + This program contains functions for motions such as orientation function and PID functions.
* motionToGoal.py
  + This program rotates robot unit it faces a desired goal then preforms motion to goal and stops motion once robot is 5 inches from the goal.
* encoder\_lib.py
  + This file includes functions of controlling the robot’s hardware including servos and encoders.
* triangulation.py
  + This program is used determine the robot location relative to 3 goals based on trilateration methods using a mapping of blob sizes to relative distance.

# Plots

### Description

To create the plots depicted above graphs the opencv blob detector was implemented on a masked image. Once the blob was recognized by the camera, a PID function was used to center the closest blob detected in the frame. Once the blob is centered within an allowable degree of error, the program implements a forward PID function moving towards the goal. The PID functions include:

# Video Analysis

To analyze the video frame and recognize and extract a single goal position for both figures, the color of each goal was determined in the HSV color space. Then the opencv library function inRange() was used to mask the frame returned from the camera. The inRange() function takes in the minimum and maximum HSV values that uniquely identify a goal. Once the goal is detected and the robot is directly facing a specific goal, the blob size is used to determine the distance of the robot from the goal. This is accomplished through a mapping of blob sizes to distance. A calibration file with blob size to relative distance was created for this purpose. With the relative distance from the goal determined by the size of the blob in the frame, trilateration methods were used to determine the x, y position of the robot.

# Link

ADD LINK HERE

# Conclusions

During the implementation of methods in this lab, our team was able to attain a deeper understanding of computer vision techniques and their application in autonomous robotics. One of the largest hurdles that we encountered was the ability to recreate the set up for conducting this lab. Specifically, we learned that if side walls were too high, the detection of blobs became more difficult when the robot was too close. This led us to have a deeper understanding of how the size of the blob was important for accomplishing the tasks in this lab. Our second attempt at the maze construction allowed us to find blobs and ascertain distances more effectively. Another issue we encountered was finding the correct lighting conditions in which the goals could be identified.