Lab 2 - PID

Austin Holmes – U#31858364

Control of Mobile Robots – Section 001F23

INTRODUCTION

Lab 2 of the Control of Mobile Robots series navigates greater depths of sensor utilization in the context of motion control. The operational paradigm used is shifted from open-loop kinematics to closed-loop "proportional, integral, and derivative" (PID) control. This control method is demonstrated by way of dynamic speed and differential drive control according to observable obstacle distances such that the robot can brake and stop before striking an object or follow walls at a predetermined distance.

TASK 1 - PID MOTION CONTROL

The first task given specified that the robot should move through a corridor, beginning with an orientation which may not be parallel to the wall(s), smoothly slowing and stopping at a specified distance from the first observable forward wall. This task is achieved principally by two control tasks, each being speed adjustment by forward distance, and differential drive control by horizontal distance.

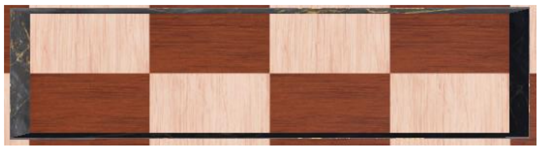


Figure 1 – Task 1 Experimental Environment

TASK 2 - WALL FOLLOWING

The second task requires a more advanced wall distance maintenance algorithm which should be able to follow a specified wall (left or right) in perpetuity or until receiving a control signal of some kind which interrupts the wall following process. This algorithm must, especially in the grid-cell-based experimental environment, have the ability to recognize and maneuver accurately through corners, or, more precisely, contexts where the wall being followed is no longer immediately detectable.

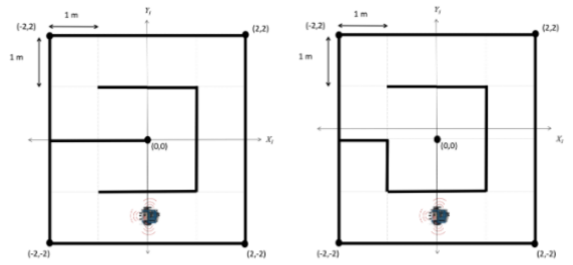


Figure 2 – Task 2 Experimental Environments

KEY VALUES

The implemented controller(s) rely on several adjustable parameters which are set upon initialization. These parameters affect all manner of sensory processing and behavior, and are critical to the functionality of the robot. This list is not exhaustive, and further omits entries which were originally relevant to prior labs.

A. Range Width

Range width specifies the total LIDAR sweep angle which is used to determine the distance reading for an input angle. For example, with the range width of ten (10), a reading for the angle ninety (90) degrees accounts for data from eighty-five (85) degrees through ninety-five (95) degrees.

B. PID "k" Constants

The PID "k" constants, utilized here both in the form of "kp" (for forward and general calculations) and "ks" (for side wall calculations), act as multipliers for calculations which determine the behavior of the robot based on error in sensor readings relative to their preset maintenance values. Higher values result in more aggressive corrections.

C. Wall Following Speed

Wall following speed simply controls the maximum speed of the outermost wheel when executing turning to correct wall distance. The ratio between this value and the "ks" constant determines the turning aggression during wall following.

D. Wall Distance Precision

Similar to the function of linear and angular precision preferences, this defines the threshold which, once exceeded, warrants corrective action for wall distance.

KEY FUNCTIONS

The overall behavior of the robot is achieved primarily by several versatile controller functions. This list is not exhaustive and omits low-level perception functions as well as high-level scripts.

A. pid_speed(error)

This function accepts the forward error (the distance between the robot and the forward wall, minus the intended distance) and outputs the allowable approach speed. Crucially, this function applies the maximum possible velocity as long as the error distance exceeds the braking distance preference, then uniformly decreases velocity according to the position of the error within the braking distance range. When within this range, the speed is calculated as follows, where v is the output velocity, v_{max} is the maximum motor velocity, d is the input error, and d_{brake} is the braking distance preference:

$$v = \mathrm{v}_{\max} \frac{d}{d_{brake}}$$

B. pid_turn(error)

Implemented for both right and left walls, this function controls differential drive depending on the error in the distance between the robot and the wall and the preferred wall distance. This function also depends on the "ks" PID constant, setting the outer wheel to the wall following speed preference and setting the inner wheel to the following, where v_{inner} is the output inner wheel velocity, v_{outer} is the outer wheel velocity (wall following speed preference), d is the input error, and k_s is "ks":

$$v_{inner} = v_{outer} - dk_s$$

C. align_wall(error)

This function precisely aligns the robot parallel to the preferred wall at the preferred distance by executing two counter-angular arching movements with radii equal to half of the input wall error. When complemented by an angle correction function, this makes up a very precise algorithm for wall following, but has been implemented in limited cases to maintain the spirit of the project. For completeness, the full arithmetic involved follows.

Arching radius, where r is the radius and d is the input distance error:

$$r = \frac{d}{2}$$

This radius is followed for ninety (90) degrees.

Angle correction, where a is the angle of the robot relative to the wall, Δd is actual difference in wall distance before and after the arching motions, and r is the radius of the correction arches:

$$a = \arcsin\left(\frac{\Delta d - 2r}{2^{\frac{3}{2}} r}\right)$$

It is also of note that this angle correction also handles corners and non-linear walls by nature, since large voids will result in potent turns toward the void, and walls which close in on the position of the robot will result in turns away from the wall.

Original work regarding the derivation of this formula can be found in ADDENDUM A but is not important to this report.

PID FUNCTIONS

Both major task behaviors are accomplished through the use of two (2) primary PID-based motion functions, each accepting its own "k" constant. As this controller acts in a simulated environment with discrete timesteps, it is critical that a sensory phase is executed between each individual execution of one of these functions.

A. move_through(distance)

This function is responsible for completing Task 1 – corridor traversal with PID speed control. During each iteration of this function, the distances to the side walls are checked, and corrections are made via *pid_turn* if the error(s) exceed the wall distance precision preference. If no correction is necessary, forward motion occurs instead. During both of these motions, the maximum speed is determined by *pid_speed*, allowing for a graceful stopping procedure. Additionally, this function will only execute if the forward error exceeds the linear precision preference, ensuring that there is a stopping condition to prevent infinite infinitesimal corrective action.

B. follow_wall(wall, distance)

This function controls the behavior specified by Task 2 – wall following. This behavior is handled by two primary states: wall following and corner handling.

During wall following, the distance from the robot to the wall is constantly maintained by *pid_turn* for the specified wall. In this case, direct forward motion is never used as the angle of the robot is highly variable, potentially leading to the introduction of larger error values. When the preferred side wall appears to be at least 2.5 times the preferred wall distance away, or the forward wall comes within the preferred wall distance, corner handling will be undertaken.

Corner handling is responsible for recognizing maze corners and acting accordingly to maintain stable wall following behavior. When encountering a corner as described above, a forward movement of the preferred wall distance divided by two (2) will be performed if the corner was recognized based on void space, and a ninety (90) degree turn will be executed. After executing the turn, the robot will move forward twice the preferred wall distance to ensure a wall is encountered, and a wall distance correction will be performed by *align_wall*, effectively "calibrating" the position of the robot to facilitate continued wall following.

NAVIGATION SCRIPT

As this lab does not require a specific path to be followed or recognized, the upper-level navigation script simply executes the relevant PID function and advances time.

CHALLENGES

The principal challenges in this lab arose from the tuning of wall following aggression by setting "ks" and handling corners in a way which allowed for continued wall following. The tuning of "ks" was primarily performed according to the idea that if the robot struck a wall while or shortly after turning toward it, or struck a wall at an angle nearer its front, the constant should be reduced. This concept was extended in kind for reduction of the constant in cases where the wall was struck at shallow angles while turning away from it. The problems regarding corner handling were addressed by way of constant linear motions which allowed the robot to avoid positions where no wall was visible. Without the motions described in CORNER HANDLING, the robot would often begin circling within corner cells, rotating erroneously several times in a row, or begin following the wrong wall or traveling in the wrong direction.

CONCLUSION

PID control, while it is highly versatile, is highly sensitive to specific application and is likely to require tuning for the machine in which it is implemented as well as the environment in which it will operate. Additionally, in an effective implementation of a closed-loop control system, live, single-step reactions to sensory data must be supplemented by precisely calculated movements such that the available data is taken full advantage of and the optimal behavior is produced.

I believe it may be optimal to fully rely on the arc-based wall following algorithm in the future.

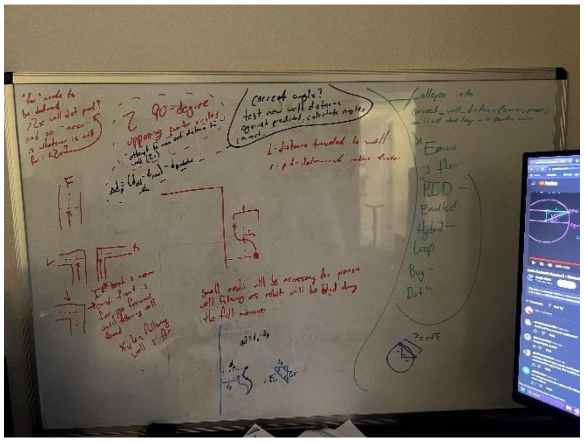
NOTES

For morale reasons, the robot class responsible for housing its control behaviors has been renamed to “Emoo,” an amalgam of a semi-phonetic alphabetic interpretation of a potential pronunciation of the abbreviation for “inertial measurement unit” (IMU) and the pronunciation of the name of the Emu bird.

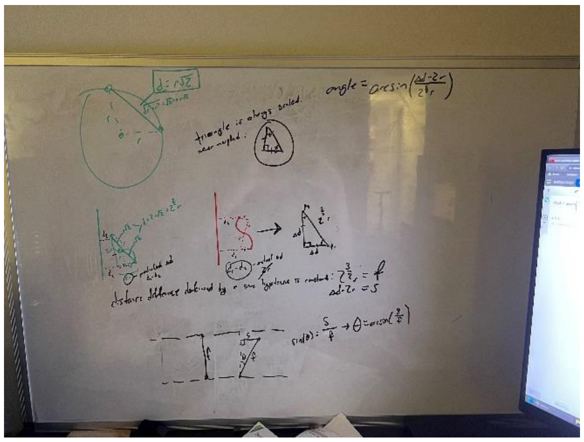
VIDEOS

Included in submission archive.

ADDENDUM A



Wall alignment motion derivation



Wall angle correction derivation