MEM380: Applied Autonomous Robots I

Wk 7 Assignment

Due: Monday, Nov. 12, 11:59pm EST

1 Objective

In this exercise, you are tasked to program and implement an open-loop and a closed-loop motion controller for the P2AT robot in USARSim . The controller will be first be implemented in MATLAB .

2 Kinematic Model of a differential-drive robot.

Let v and ω denote the robot's forward speed and angular turn rate respectively. Recall that v and ω can be written as functions of the left and right wheel speeds of the robot, $\dot{\phi}_L$ and $\dot{\phi}_R$, the wheel radius, r, and one half the wheel base, l as follows:

$$v = \frac{r(\dot{\varphi}_R + \dot{\varphi}_L)}{2}, \tag{1a}$$

$$\omega = \frac{r(\dot{\varphi}_R - \dot{\varphi}_L)}{2l}. \tag{1b}$$

Given v and ω, we can describe the robot's kinematics as follows:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} v \\ \omega \end{pmatrix} \tag{2}$$

where x, y, and θ denotes the robot's pose (position and orientation) in the intertial frame (see Figure 1. In USARSim , we will be using the INS sensor to get the robot's global positions and orientations.

Task 0: Download the Matlab files for this assignment. Make sure you have the following files:

- `differentialDriveMain_user.m` - This is the main script for this assignment
- `angleDifference.m` - Helper function used to compute the difference between two angles. This function takes angles in the range of $[0, 2\pi)$ and outputs an angle difference also in the range of $[0, 2\pi)$.
- `plotRobotTrajectory.m` - Helper function for plotting the resulting trajectory.

Before you begin, you should look over each file. Read over the comments and make sure you can locate all the sections you are expected to contribute code.

What to turn in: In your submission e-mail, write a brief 1 paragraph description on the functionality of the `differentialDriveMain_user.m` and how it uses the other helper functions to achieve its purpose.

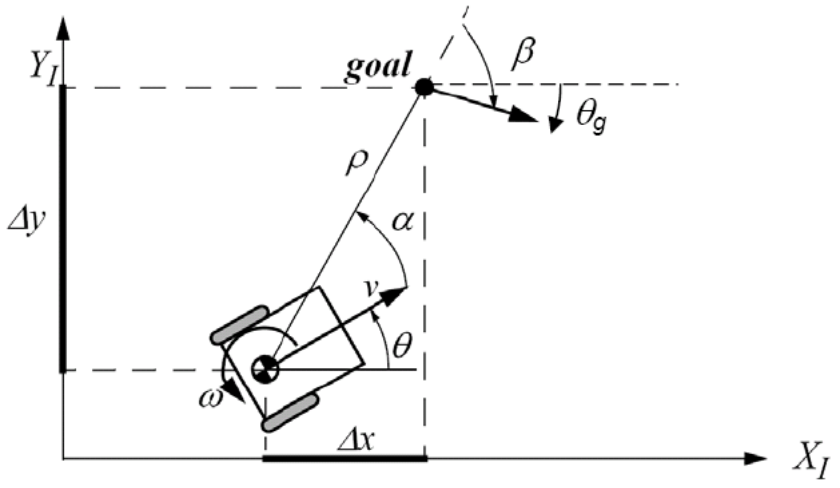

Figure 1: Schematic for closed-loop motion control.

3 Open-loop control: assign speed commands to the robot

Task 1: Your first task is to adapt the MATLAB `differentialDriveMain_user.m` function to enable your robot to drive from a start pose of $[-5\ -5\ 1.8\ 0\ 0\ 0]^T$ to a goal pose of $[0\ 0\ 1.8\ 0\ 0\ \pi/2]^T$. To achieve this, you mist determine the sequence of v and ω control commands and store each in `vOpenLoop` and `omegaOpenLoop` respectively (see Lines 43-44 `differentialDriveMain_user.m`).

To enable your robot to move, you must complete the code in `differentialDriveMain_user.m` (see Lines 61-69 and possibly Lines 87-90). Once you have completed this, run the script `differentialDriveMain_us` to see the resulting trajectory.

What to turn in: Rename `differentialDriveMain_user.m` to `differentialDriveMain_user_T1.m` where you should replace `user` with your Drexel domain login. For example, my Drexel domain login is $mah349$ and so I would rename the files `differentialDriveSimulator_mah349_T1.m`.

4 Closed-loop Control

Task 2: For a precise and smooth movement of the robot, we propose a linear control law that takes into account how far away the robot is to its goal location and the difference between its current orientation and its final orientation. In other words, we set v and ω as follows:

$$v = k_\rho \rho \tag{3a}$$

$$\omega = k_\alpha \alpha + k_\beta \beta \tag{3b}$$

where ρ, α, and β are shown in Figure 1. Here k_ρ, k_α, and k_β are called controller gains and constants that we select to enable the robot to reach the desired goal configuration. To ensure convergence and stability of the resulting closed-loop robot motion, these controller gains must be chosen to satisfy the following conditions:

(a) $k_\rho > 0$;

(b) $k_\beta < 0$; and

(c) $k_\alpha + (5/3)k_\beta - (2/\pi)k_\rho > 0$.

Closed-loop implementation:

- Specify the start and goal pose in Lines 19-20 of `differentialDriveMain_user.m`.
- Compute the ρ, α, and β using the robot's current pose, `robPose`, and the goal pose, `goalRobPose` (see Lines 71-77 of `differentialDriveMain_user.m`).
- Set your controller gains, K_ρ, K_α, and K_β in Lines 46-52 of `differentialDriveMain_user.m`.
- Use ρ, α, β, K_ρ, K_α, and K_β to determine v and ω in Lines 76-77 `differentialDriveMain_user.m`.
- Test the controller for different positions and orientations by running `differentialDriveMain_user.m`.
- You may have to try a few different controller gains to achieve the desired results. **Hint:** The magnitude of K_ρ, K_α, and K_β are small for this to work. If you chose magnitudes that are too large, *i.e.* > 1), your robot may simply drive in a straight line.

Tips:

- Use `atan2` when computing angles.
- Use `mod(atan2(dy, dx), 2*pi)` to obtain angles in the range of 0 and 2π
- If you are not sure what a function does, type `help functionName` on the Matlab command prompt. This works for all the helper functions in this assignment.

What to turn in: Rename `differentialDriveMain_user.m` to `differentialDriveMain_user_T2.m` where you should replace `user` with your Drexel domain login. Zip up all your files and attach it to your submission e-mail.