Peter Huynh

September 16, 2016

CSCI 3302

Lab 2.2

Inverse Kinematics/Control Workshop

Goal:

- To understand how to calculate the wheel speed given a desired velocity
- Experience basic feedback control
- See the need for higher-level reasoning / path-planning

Required:

- Sparki robot with working odometry code
- Line-following course
- U-Shaped obstacle

Instructions:

1. Use the forward kinematic relationship for a differential wheel platform from the book (x_r' and theta_r' as a function of left and right wheel speed) to calculate its inverse, that is left and right wheel speeds given x_r' and theta_r'.

   To calculate both individual wheel speeds, we used Page 64's Equation 3.64 to solve for immediate speed of the wheel at a specific moment.

$$\begin{aligned} \dot{\phi}_l &= (2\dot{x}_R/r - \dot{\theta}d)/2 \\ \dot{\phi}_r &= (2\dot{x}_R/r + \dot{\theta}d)/2 \end{aligned}$$

   Where r is the radius of wheel, 0.0254 meters, and d is the distance of the axis between the two wheels, 0.0841375 meters.

   x_r' and theta_r' were both calculated using lab 2.1's odometry code. Both represent forward and rotational velocity at the given moment.

2. Implement a feedback controller that drives to a specific goal provided by coordinates X, Y, and THETA, where THETA is the final orientation of the robot by doing the following:
   - Calculate the Euclidian distance RHO between your current location and the goal

     RHO was determined using this equation: (Page 65, Equation 3.65)

     $$\rho = \sqrt{(x_r - x_g)^2 + (y_r - y_g)^2}$$

     x_r and y_r are both determined in the odometry code, both zero at startup and changing with each loop iteration, while x_g and y_g are preset constant values.
   - Calculate the angle ALPHA that the goal is oriented at. For example, if the goal is to the robot's right, ALPHA is 90 degrees, if it's in front, its 0 degrees and so on. You should use the function atan2() for doing this.

     ALPHA was calculated using this equation: (Page 65, Equation 3.65)

     $$\alpha = \theta_r - \tan^{-1} \frac{y_r - y_g}{x_r - x_g}.$$

     θ_r is the robot's overall angle compared to the map, and all y and x values are the same ones used in RHO. We used atan2() to ensure correct arctangent calc.
   - Calculate a forward speed x_r' that is proportional to RHO, e.g. x_r' = 0.1*RHO

     Using the equation below (Page 65, Equation 3.66):

     $$\dot{x} \qquad = p_1 \rho$$

     We set p_1 to be 0.1 inside out calculations in the code to determine forward speed. This will now be the value used to determine separate wheel velocities
   - Calculate a rotational speed theta_r' that is proportional to ALPHA and THETA, e.g. theta_r'=0.1*(ALPHA-theta_r)+0.01*(THETA-theta_r) and make sure the direction of theta is consistent with your coordinate system.

     Using these two equations (Page 65, Equations 3.65 and 3.67):

     $$\dot{\theta} \quad = p_2 \alpha + p_3 \eta$$

     $$\eta = \theta_g - \theta_r$$

     We set p_2 and p_3 tp 0.1 and 0.01 respectively, and apply that to our code calculations. This new rotational velocity will be used for wheel speeds.

Test your controller by letting the robot drive to random locations on the line following map. (Make sure you track the motion commands in your odometry code.) Provide the equations for the final controller in your writeup. Answer the following: What happens if you decrease your constants (here 0.1)? What happens if you increase them? What happens if they get too big? What is the role of the 0.01*(THETA-theta_r) term?

> Decreasing the p constants reduce the overall speed of the forward and rotational velocities. These reductions have different effects on the separate wheels. If forward velocity decreases (p_1), both wheel's overall speed decreases. However, if rotational velocity decreases (p_2 or p_3), the difference in wheel speed decreases, meaning the robot will travel more straight.
>
> Increase the p constants have the adverse effect on the speed of the wheels. Increased forward means overall speed increase, while increased rotation causes a larger difference and more angular movement.
>
> If both reach too high of a value, the wheel speed and differences would be too unreasonable to motor calculations. Sparki would not move, but the calculations done would constant spiral larger and larger. Usually, if the value is too big, values approach infinity and are dropped to nan.

3. Feel free to implement obstacle avoidance using the ultrasound sensor. (Not obligatory.) Assume the robot is able to turn away from obstacles while executing your controller. What happens if you place a u-shaped obstacle between the robot and its desired location?

> Optimally, if a robot with object avoidance were to approach a u-shaped obstacle, similar to a dead end in a maze, the robot would first enter the center of the u-shape and then realize that there is an obstacle to avoid. It would then track the object's edges, following along the u-shape ends until it leaves the center. After exiting the center, it's a simple matter to traverse past one corner edge and then moving back into the same direction prior to encounter the u-shape. The robot is now not trapped in the center and passes the obstacle on one of its sides.