Lab 3: Inverse Kinematics Hints

CSCI 3302: Introduction to Robotics

Part 2:

Your loop() code should consist of a TURN-MOVE-TURN.

First, you'll eliminate your bearing error.

(Once you make this first turn, you'll want to update your pose_theta and re-calculate your heading error.)

Second, you'll eliminate your distance error.

Third, you'll eliminate your heading error.

Finally, you need to update the robot pose to be the destination pose
(so next loop() call all of your errors are 0)

Part 3:

1. Update your odometry function using the equations from Lab 2. Keep in mind that the distance each wheel has moved will be scaled by the % you give to sparki.motorRotate at the end of your loop() function

2. In your loop() function: make sure you're computing your 3 errors each time

3. Choose a function to calculate $\dot{x}_R$ and $\dot{\theta}$. $\dot{x}_R$ is typically a function of ρ.
   $\dot{\theta}$ is usually a function of α and η (a trade-off between bearing and heading error),
   and sometimes ρ. An example of a $\dot{\theta}$ function using all three would be one that changes how it trades off between bearing and heading error based on how far away the robot is from its goal position.

4. Compute $\dot{\phi}_l$ and $\dot{\phi}_r$, the amount each wheel needs to rotate, using your computed values for $\dot{x}_r$ and $\dot{\theta}$.

5. Using $\dot{\phi}_l$ and $\dot{\phi}_r$, compute the speed to assign to each wheel. One strategy for doing so is to set the speed of the wheel that has longer to travel (a larger $\dot{\theta}$) to 100%, and the other wheel proportional to it.

6. Consider the case where $\dot{\phi}_l$ can be negative (if $\frac{\dot{\theta} d}{2} > \dot{x}_R$, it will be negative). If this happens, then you'll want to rotate your left wheel *backwards*. Make sure your odometry can handle this! One way is to allow your left_wheel_pct to be negative, but wrap it in an abs() calls wherever appropriate (e.g,. within your motorRotate command) to keep it positive where it needs to be.

7. Include a stop condition for your controller: if you're within some predetermined d_err and h_err, tell the robot to sparki.moveStop(), and set your left_speed_pct and right_speed_pct to 0 so the odometry doesn't erroneously update.