- (15) **Feedback control of differential drive robot** Consider a differential drive model of the mobile robot discussed in the class with a configuration space $[x, y, \theta]^T$, where we can control linear and angular velocity of the robot v and ω. The kinematic model describing the motion of the mobile robot is as follows:

$$\begin{aligned} \dot{x} &= v\cos(\theta) \\ \dot{y} &= v\sin(\theta) \\ \dot{\theta} &= \omega \end{aligned}$$

  - Implement the closed loop feedback control law for v and ω or moving to a goal point $\mathbf{x}^* = [x^*, y^*]^T$ described in the slides. `[x,y,theta] = goToPoint(`x_0`,` x_g`)`
  - Implement the closed loop feedback control law for v and ω following a line given by parameters $l = [a, b, c]^T$. `[x,y,theta] = followLine(`x_0`,` l`)`
  - Implement the closed loop feed-back control law `[x,y,theta] = goToPose(`x_0`,` x_g`)`,

$$\begin{aligned} v &= k_\rho \rho \quad (1) \\ \omega &= k_\alpha \alpha + k_\beta \beta \quad (2) \end{aligned}$$

    described in class, where ρ, α, β is the configuration of the robot expressed in polar coordinates (with respect to the goal). Experiment with different values of $k_\rho, k_\alpha, k_\beta$ and describe any observations and strategies you chose to decide whether the goal was reached.

  As a part of this exercise you can use the function written above, which simulates the odometry of the mobile robot. The control laws (functions) will keep on applying the control law and simulating the robot forward and checking the flag which indicates that the goal position is reached. It is assumed that the goal position x_g and initial position are specified in the inertial coordinate frame. Test the control law for variety of initial/goal positions. Post the plots of $x(t), y(t), \theta(t)$, the printout of the code and any observations you may have and some plots of the trajectories in x-y space for several initial conditions.

- (5) **Potential Field Based Control**. Consider a point like robot in the workspace with the area $[0; 100] \times [0, ; 100]$. Represent the obstacles in the environment as circles with centers at $[40; 30]$ and $[70; 40]$ each with radius 5. Assume that the initial position of the robot is $x_0 = [0; 0]$ and goal position is $x_g = [100; 60]$.

  Write down a function `GoToAvoid(`$\mathbf{x}, \mathbf{x_g}, \mathbf{x_o}$`)`, which takes as an input arbitrary goal position in the workspace and position of the obstacles and computes velocity vector control command. In this part the velocity vector should be computed as a sum of gradient vectors of the attractive and repulse potential function described in the slides. The additional constant parameters of the potential functions can be set as variables inside of `GoToAvoid`.

  Submit the Matlab/Python code and plots demonstrating the capability of the robot to avoid obstacles and reach the goal.