CPE 521-WS Autonomous Mobile Robotic Systems

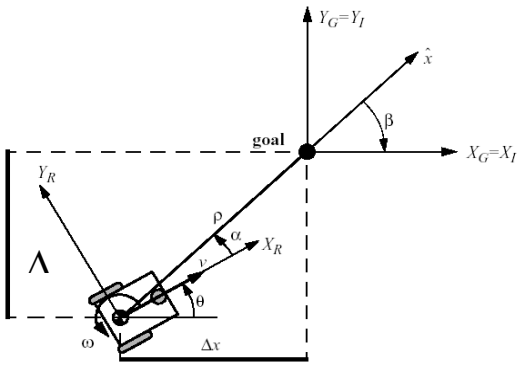
Lab 1: Matlab Simulation for Kinematic Position Control

Objective:

Simulate the kinematic position control of a differential drive robot.

The Assignment:

For a differential drive mobile robot as shown in the picture, program in Matlab to simulate the paths shown in Figure 3.20 of the Textbook, where the robot is initially on a circle in the *xy* plane. All movements should have smooth trajectories toward the goal in the center. Verify the control parameters given in equation (3.59) of the Textbook, or give the parameters that you use to generate similar trajectories.



Lab Instruction:

The kinematic model of a differential drive robot is given by:

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

The feedback control input is derived to be:

$$v = k_\rho \rho;\ \omega = k_\alpha \alpha + k_\beta \beta,$$

where $\rho = \sqrt{\Delta x^2 + \Delta y^2}$; $\alpha = -\theta + \text{atan}\,2(\Delta y, \Delta x)$; $\beta = -\theta - \alpha$, and k_ρ, k_α and k_β are constants satifying the condition: $k_\rho > 0$; $k_\beta < 0$; $k_\alpha - k_\rho > 0$. (see Lecture 4 and Textbook Section 6.2 for detailed feedback control design and stability analysis.)

Matlab Simulation:

The Matlab function "ode45" solves differential equations of the form: $\dot{y} = f(t, y)$.

>>[t,y] = ode45(odefun,tspan,y0)

where tspan = [t0 tf], integrates the system of differential equations $\dot{y} = f(t, y)$ from t0 to tf with initial conditions y0. Each row in the solution array y corresponds to a value returned in column vector t.

In our case, we can call this function:

>>[t,y]=ode45(@def_robot,[0 20],X0);

where we need to define the initial configuration of the robot, X0. The following code provides an example to put the robot initial configuration on one point of the circle with radius 50:

```
>>r=50;
>>angle=-3*pi/4;
>>x_0=r*cos(angle);
>>y_0=r*sin(angle);
>>theta_0=0;
>>X0=[x_0,y_0,theta_0];
```

We also need to define the ode function:

```
function xdot=def_robot(t,x)

%Initialize the state vector
xdot = zeros(3,1);

%Defining parameters
k_p=3;
k_alpha=8;
k_beta=-1.5;

x_f=0;
y_f=0;
theta_f=0;

rho=sqrt((x_f-x(1))^2+(y_f-x(2))^2);
alpha=-x(3)+atan2((y_f-x(2)),(x_f-x(1)));
beta=-x(3)-alpha;

%Control input
v=k_p*rho;
```

```
omega=k_alpha*alpha+k_beta*beta;

%defining the differential equations
xdot(1)=cos(x(3))*v;
xdot(2)=sin(x(3))*v;
xdot(3)=omega;
end
```

The ode45 function returns the time vector, “t”, and the state matrix, “y”. Note that “t” is a column vector, and “y” is a matrix where the first column is the value of the state variable x, the second column is the value of the state variable y, and the third column is the value of the state variable θ (i.e., theta in the Matlab code). Also note that the number of rows of the vector “t” equals the number of rows of the matrix “y”. You can check the dimensions of “t” and “y” using the following commands in Matlab:

```
>>size(t)

>>size(y)
```

To plot the x- and y- positions of the robot, you can use:

```
>>x1=y(:,1);
>>y1=y(:,2);
>>plot(x1,y1);
```

To add a red cross at the start position of the robot and a blue circle at the end position of the robot, you can use the following codes:

```
hold on

plot(x1(1),y1(1),'rx'); %plot the inital position of the robot as a red x
hold on
plot(x1(length(t)),y1(length(t)),'bo'); %plot the end position of robot as a blue o

xlabel('X');
ylabel('Y');

hold off
```

Note that when you use “hold on” to retain current plot when adding new plots, you need to finish it with “hold off” to stop holding the plots.

The above instruction shows you how to plot one curve for kinematic position control. Please follow the instruction and complete all other curves with initial conditions distributed on the circle of radius 50.

Lab Report Submission

In the lab report, please summarize the method you use, parameter values, and the simulation results (with Matlab generated figures shown in the report). You may use any document preparation software such as Word or Latex, and save the file as a single pdf file for submission. Please follow the Lab Report Submission link in Canvas to submit the lab report.