# Principles of Robot Autonomy I
# Homework 2 SOLUTIONS
# Due Tuesday, October 10 (5pm PT)

## Problem 1

(i) Solution in code. For discussion, the states are $(x, y, \theta)$ and control inputs are $(v, \omega)$. The trajectories show that $(x, y)$ have a constant slope due to the constant velocity input and $\theta$ oscillates since $\omega$ is a sinusoidal input.

(ii) $\mathbf{X}_{t+1} = \bar{\mathbf{A}}\mathbf{X}_t + \bar{\mathbf{B}}\mathbf{u}_t$ Note: Any notation for the control vector is fine, as long as it is defined.

(iii) Solution in code. For discussion, the main difference between the two models is that the state in the DoubleIntegrator model includes the velocities, $(v_x, v_y)$ and the control inputs are virtual controls -> $(a, \omega)$.

## Problem 2

(i) We wish to choose the coefficients for the basis functions that make up the trajectory such that the boundary conditions are satisfied. We have eight unknowns (4 for $x$, 4 for $y$), and eight conditions (4 initial, 4 final), so this is possible. Our constraint equations for $x(t)$ are:

$$\begin{aligned} x(0) &= 0 = x_1 \\ x(t_f) &= 5 = x_1 + x_2 t_f + x_3 t_f^2 + x_4 t_f^3 \\ \dot{x}(0) = v(0)\cos(\theta(0)) &= 0 = x_2 \\ \dot{x}(t_f) = v(t_f)\cos(\theta(t_f)) &= 0 = x_2 + 2x_3 t_f + 3x_4 t_f^2 \end{aligned}$$

We can write this in matrix form as:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & t_f & t_f^2 & t_f^3 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2t_f & 3t_f^2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 5 \\ 0 \\ 0 \end{bmatrix}$$

A similar analysis of the $y$ coefficients (replacing cos with sin) yields the following matrix equation:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & t_f & t_f^2 & t_f^3 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2t_f & 3t_f^2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 5 \\ -0.5 \\ -0.5 \end{bmatrix}$$

These equations can be solved to find the coefficients for $t_f = 25$.

(ii) The matrix that defines the virtual control inputs:

$$\begin{bmatrix}\ddot{x}(t)\\ \ddot{y}(t)\end{bmatrix} = \underbrace{\begin{bmatrix}\cos(\theta) & -v(t)\sin(\theta)\\ \sin(\theta) & v(t)\cos(\theta)\end{bmatrix}}_{:=J}\begin{bmatrix}a\\ \omega\end{bmatrix} := \begin{bmatrix}u_1\\ u_2\end{bmatrix}.$$

is no longer invertible ($\det(J) = v = 0$). We cannot define the velocities, accelerations, etc. in terms of the real control inputs, so the system is not differentiably flat.

(iii) Solution in code.

(a) Any correct expression for $\theta$ in terms of the states is allowed. Code solution uses

$$\theta = atan(\frac{\dot{y}}{\dot{x}})$$

(b) Trajectory starts and ends with $\theta = -\pi/2$, causing the turtlebot to face downward and then curve upward. Open loop is subject to propagation errors (or noise) since there is no error correction so can divert from nominal traj.

(iv) Solution in code

(v) Using $\dot{v}(t) = a(t)$, $\dot{\theta}(t) = \omega(t)$, the virtual controls $u_1$ and $u_2$, are related to the true control inputs through the following equation:

$$\begin{bmatrix}\ddot{x}(t)\\ \ddot{y}(t)\end{bmatrix} := \begin{bmatrix}u_1(t)\\ u_2(t)\end{bmatrix} = \begin{bmatrix}cos(\theta(t)) & -v(t)sin(\theta(t))\\ sin(\theta(t)) & v(t)cos(\theta(t))\end{bmatrix}\begin{bmatrix}\dot{v(t)}\\ \dot{\theta(t)}\end{bmatrix}$$

If $v(t) \neq 0$ then $\begin{bmatrix}cos(\theta(t)) & -v(t)sin(\theta(t))\\ sin(\theta(t)) & v(t)cos(\theta(t))\end{bmatrix}$ is invertible and the true control is given by the following equation:

$$\begin{aligned}\begin{bmatrix}\dot{v}(t)\\ \dot{\theta}(t)\end{bmatrix} &= \begin{bmatrix}cos(\theta(t)) & -v(t)sin(\theta(t))\\ sin(\theta(t)) & v(t)cos(\theta(t))\end{bmatrix}^{-1}\begin{bmatrix}u_1(t)\\ u_2(t)\end{bmatrix}\\ &= \frac{1}{v(t)}\begin{bmatrix}v(t)cos(\theta) & v(t)sin(\theta(t))\\ -sin(\theta(t)) & cos(\theta(t))\end{bmatrix}\begin{bmatrix}u_1(t)\\ u_2(t)\end{bmatrix}\end{aligned}$$

(vi) Solution in code.

(a) using np.linalg.solve or using $J^{-1}$

(b) It doesn't perfectly track the nominal since it is only using PD control. It tracks much better than the open-loop control.

# Problem 3

(i) 6 states.

(ii) 2 controls.

(iii) Constrained, non-linear trajectory optimization solved with a direct method.

(iv) Solution in code. (a) 2*6 (b) Q matrix or R matrix