COMP7308 Introduction to unmanned systems 2021

Assignment 1

:

1 2D transformations as Affine Matrices (10 points)

Transformations between coordinate frames play an important role in robotics. As background for this part, please refer to Lecture 2.

Considering a robot moving on a plane, its pose w.r.t. a global coordinate frame is commonly written as $\mathbf{x} = (x, y, \theta)^T$, where (x, y) denotes its position in the xy-plane and θ is its orientation. The homogeneous transformation matrix that represents a pose $\mathbf{x} = (x, y, \theta)^T$ w.r.t the origin $(0, 0, 0)^T$ of the global coordinate system is given by

$$\mathbf{T} = \begin{pmatrix} \mathbf{R}(\theta) & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}, \ \mathbf{R}(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, \ \mathbf{t} = \begin{pmatrix} x \\ y \end{pmatrix}.$$

(a) (2.5 points) While being at pose $\mathbf{x}_1 = (x_1, y_1, \theta_1)^T$, the robot senses a landmark l at position $\mathbf{l} = (l_x, l_y)$ w.r.t its local frame. Please calculate the coordinates of l w.r.t the global frame.

(b) (2.5 points) Now imagine that you are given the landmark's coordinates w.r.t. the global frame. How can you calculate the coordinates that the robot will sense in its local frame?

(c) (2.5 points) The robot moves to a new pose $\mathbf{x}_2 = (x_2, y_2, \theta_2)^T$ w.r.t. the global frame. Find the transformation matrix ${}^1_2\mathbf{T}$ that represents the new pose w.r.t. to the original pose $\mathbf{x}_1$. Hint: the result should be a product of homogeneous transformation matrices.

(d) (2.5 points) The robot is at position $\mathbf{x}_2$. Where is the landmark $\mathbf{l} = (l_x, l_y)$ w.r.t. the robot's local frame now?

2 Sensing (10 points)

A robot is located at $x = 1.0\,\text{m}, y = 0.5\,\text{m}, \theta = \frac{\pi}{4}$. Its lidar is mounted on the robot at $x = 0.2\,\text{m}, y = 0.0\,\text{m}, \theta = \frac{\pi}{2}$ w.r.t. the robot's local frame. The distance measurements of one lidar can be found in the file `laserdata.bat`, which is in the attachment. The first distance measurement is taken in the angle $\alpha = -\frac{\pi}{3}$ (in the frame of the lidar sensor), the last distance measurement has $\alpha = \frac{\pi}{3}$ (i.e. the field of view of the sensor is $\frac{2}{3}\pi$), and all neighboring measurements are in equal angular distance (all angles in radians).

Note: You can load the data file and calculate the corresponding angles in Python using

```
import math
import numpy as np
scan = np.loadtxt('laserscan.dat')
angle = np.linspace(-math.pi/3, math.pi/3, np.shape(scan)[0], endpoint='true')
```

(a) (2.5 points) Use Python to plot all laser end-points in the frame of reference of the lidar sensor.

(b) (2.5 points) The provided scan exhibits an unexpected property. Identify it and suggest an explanation.

(c) (5 points) Use homogeneous transformation matrices in Python to compute and plot the center of the robot, the center of the lidar sensor, and all lidar end-points in the world coordinates.

Note: you need to equally scale the $x-$ and $y-$axis of a plot using

```
plt.gca().set_aspect('equal', adjustable='box')
```

3 Two-wheeled robot (20 points)

We have discussed the two-wheeled robot in Lecture 3, where we derive its continuous motion model as

$$\begin{pmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{pmatrix} = \begin{pmatrix}v\cos\theta\\ v\sin\theta\\ \omega\end{pmatrix} = \begin{pmatrix}\frac{r\omega_1+r\omega_2}{2}\cos\theta\\ \frac{r\omega_1+r\omega_2}{2}\sin\theta\\ \frac{r\omega_1-r\omega_2}{2l}\end{pmatrix}.$$

There are many ways to discretize this motion model, and one popular way is as follows:

$$\begin{pmatrix}x'\\ y'\\ \theta'\end{pmatrix} = \begin{pmatrix}\cos(\omega\Delta t) & -\sin(\omega\Delta t) & 0\\ \sin(\omega\Delta t) & \cos(\omega\Delta t) & 0\\ 0 & 0 & 1\end{pmatrix} \cdot \begin{pmatrix}x-\mathrm{ICR}_x\\ y-\mathrm{ICR}_y\\ \theta\end{pmatrix} + \begin{pmatrix}\mathrm{ICR}_x\\ \mathrm{ICR}_y\\ \omega\Delta t\end{pmatrix}.$$

As we explained in class, ICR is the rotation center of the robot and its coordinate can be computed as $\mathrm{ICR} = \begin{pmatrix}x+R\sin\theta\\ y-R\cos\theta\end{pmatrix}$. $R=\rho-l$ is the rotation radius of the robot around the ICR, and $\omega = \dfrac{(\omega_1-\omega_2)r}{2l}$ is the rotation rate. Here l, r and ρ are the symbols we used in lecture to derive the motion model. We also assume the wheel radius $r = 1\,\mathrm{m}$.

(a) (2.5 points) After some derivation, we can get $R = l\dfrac{\omega_2+\omega_1}{\omega_2-\omega_1}$. Explain what will R be when $\omega_2 = \omega_1$ and why.

(b) (5 points) Write a function in Python that implements the motion model for the two-wheeled robot using the discrete motion model explained above. The function header should look like `function [x_n y_n theta_n] = diffdrive(x, y, theta, w_1, w_2, t, l)`.

(c) (2.5 points) After reaching position $x = 1.5\,\mathrm{m}$, $y = 2.0\,\mathrm{m}$, and $\theta = \pi/2$ the robot executes the following sequence of steering commands:

1. $c_1 = (\omega_1 = 0.2\,\mathrm{rad/s}, \omega_2 = 0.3\,\mathrm{rad/s}, t = 3s)$
2. $c_2 = (\omega_1 = -0.1\,\mathrm{rad/s}, \omega_2 = 0.1\,\mathrm{rad/s}, t = 1s)$
3. $c_3 = (\omega_1 = 0\,\mathrm{rad/s}, \omega_2 = 0.2\,\mathrm{rad/s}, t = 2s)$

Use the function to compute the position of the robot after the execution of each command in the sequence (the distance $2l$ between the wheels of the robot is $0.5\,\mathrm{m}$).

(d) (2.5 points) A two-wheeled robot starts at position $x = 1.0\,\mathrm{m}$, $y = 2.0\,\mathrm{m}$ and with heading $\theta = \frac{\pi}{2}$. It has to move to the position $x = 1.5\,\mathrm{m}$, $y = 2.0\,\mathrm{m}$, $\theta = \frac{\pi}{2}$ (all angles in radians). What is the minimal number of steering commands (ω_1, ω_2, t) needed to guide the vehicle to the desired target location?

Note: according to the discretized motion model, we know that given a fixed steering command, the robot is rotating around the ICR.

(e) (2.5 points). There are many ways to use the minimal number of steering commands to accomplish (d), and each of them will result in a different path. Among all these paths, what is the length of the shortest one?

(f) (5 points) Please change `diffdrive` function to add disturbance to the robot. In particular, the function inputs should include the noise levels for different state elements, and the function output should be a set of samples of the next time step states. You need to implement two different ways to add the disturbance: (i) additive Gaussian noise and (ii) directly add the noise to the current state elements. Please use your new functions to plot the results for (c), which should be similar to the sequence of point clouds that we demonstrated in class. Please discuss the phenomena you observed from the results (e.g., when you use different choices for the noise, you may observe the banana shape distribution or Gaussian distribution).

4 Dynamic programming for optimal control (30 points)

Consider the one-dimensional system

$$x_{k+1} = x_k + u_k + w_k, \quad k = 0, 1, 2, 3$$

with initial state $x_0 = 5$ and the cost function

$$\sum_{k=0}^{3} (x_k^2 + u_k^2).$$

Apply the dynamic programming algorithm for the following two cases:

(a) (15 points) We assume the control actions can only be selected from the constraint set $U_k(x_k) = \{u \mid 0 \leq x_k + u \leq 5, u : \text{ integer}\}$ for all x_k and k, and the disturbance w_k is equal to zero for all k. Please compute the optimal x_k and u_k sequence.

(b) (15 points) We now make a different assumption: the control constriant set is as in (a), but the disturbance w_k now takes the values -1 and 1 with equal probablity $1/2$ for all x_k and u_k, except if $x_k + u_k$ is equal to 0 or 5, in which case $w_k = 0$ with probability 1. Please compute the optimal action u_0 and the optimal cost. Do you think we can compute the optimal action sequence and state sequence? If yes, please give your result. Otherwise, please explain why.

You can either solve these two questions using python or doing the computation on the paper. If you choose to use python, please submit your code and write down the optimal sequence in the paper. If you choose to solve it by hand, please list the detailed computation steps.

5 State-space formulation of the an affline system (10 points)

In class we mainly focus on the "standard" state-space formulation of a linear system:

$$\dot{x} = Ax + Bu$$
$$y = Cx$$

Given the following "non-standard" state-space formulation of an affine system:

$$\dot{x} = Ax + Bu + \Delta$$
$$y = Cx,$$

where Δ is a non-zero constant. Please convert it into the standard state-space formulation. (Hint: let $\tilde{x} = \begin{bmatrix} x \\ 1 \end{bmatrix}$.)

6 Reading, Discussing and Implementing a Paper (20 points)

Please read the paper "Humanoid Full-Body Manipulation Planning with Multiple Initial Guesses and Key Postures, Tang et al., 2015". The pdf paper can be found in the homework folder.

(a) (5 points) Please summarize the challenges that this paper has solved.

(b) (5 points) Please summarize the main techniques used in the solution provided in this paper. Please be concise – list a few keywords shall be sufficient.

(c) (10 points) Implement a simple version of the inverse kinematics algorithm proposed in this paper (details in III.C of the paper) for a 2D link robot similar to the one shown in Figure 3 in this paper. You may consider to add one or two box shaped obstacle in the scene. For the optimization solver, you may consider the `scipy.optimize.minimize` by choosing the `SLSQP` solver.

(Hint: this is an open-ended question and we will assess your answer according to your efforts and solution quality).