

Homework1

For the homework, I designed the algorithm on the paper, so I attached the images here.

CSE276A

x	y	angle
0	0	0
-1	0	0
-1	1	1.57
-2	1	0
-2	2	-1.57
-1	1	-0.78
0	0	0

Suppose:

at the beginning $(\theta=0)$ is the north frame for this task.
our robot is located at the origin and the orientation is same as the direction of axis x . ($\theta=0$)

Using the bicycle model of the four-wheels car, he could get the \dot{x} , \dot{y} and $\dot{\theta}$ as follows:

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

and the turning angle for the front wheels γ is:

$$\gamma = \tan^{-1} \frac{w \cdot L}{v} \quad (L \text{ is the distance between front and back wheels})$$

Then, we use polar coordinate to measure the distance and angle between the ~~goal~~ goal position and the current position.

$$\begin{cases} P = \sqrt{x^2 + y^2} \\ \alpha = \tan^{-1} \frac{y}{x} - \theta \\ \beta = -\theta - \alpha + \beta^* \end{cases}$$

β^* is the angle of the next point
 θ is the angle between the robot orientation and the x axis.

For example, from the beginning point $(0,0,0)$ to the next point $(1,0,0)$,
 θ is equal to 0 and x is 1, y is 0, α is 0. β is 0.

$$\begin{cases} \theta=0 \\ P=1 \\ \alpha=0 \\ \beta=0 \end{cases}$$

from the point $(-1,0,0)$ to the point $(-1,1,1.57)$

$$\begin{cases} \theta=0 \\ P=1 \\ \alpha=\frac{\pi}{2} \\ \beta = \frac{\pi}{2} - \frac{\pi}{2} = 0 \end{cases}$$

And then, we could set three parameters $k_p, k_{\alpha}, k_{\beta}$, let:

$$\begin{cases} v = k_p \cdot p \\ w = k_{\alpha} \cdot \alpha + k_{\beta} \cdot \beta \end{cases} \quad k_p > 0, k_{\beta} < 0, k_{\alpha} - k_p > 0.$$

if we know the $k_p, k_{\beta}, k_{\alpha}$, Then we could get the desired v and w .

Once v and w are known, we could control the angular velocity of the back wheels ($\omega_r = \omega_l = \frac{v}{r}$) and the turning angle of the front wheels $\gamma = \tan^{-1} \frac{w}{v}$.

If we could use some kind of sensors to measure the orientation of the robot during running, we could adjust the k_{α} and γ at real time.

To illustrate the performance of my algorithm, I used MATLAB for simulation and I put the video and the m file in the zip file which shows the results. In this case, the control parameters are as follows:
 $k_p = 0.08$;

```
ka=0.09;  
kb=-0.11;
```

When I run the algorithm in python on the PiCar platform, I find for open-loop control system, I could not adjust the turning angles in real time. Hence, I just designed a open-loop control system for PiCar to pass through all the points as required. From the video, you can see that the car has successfully passed through all the points. Besides, I attach the python code in the zip file.