1.A report that describes your solution

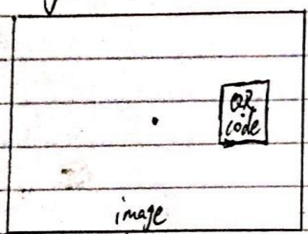
HW2 CSE 276A Yunhai Han

1. A report that describes your solution

In the homework2, we are allowed to take advantage of a webcamera on picar to improve the localization performance.

Compared with the open-loop algorithm in homework1, the feedback control algorithm could know their localization relative to the QR codes. And, since we know these QR codes' localization beforehand (we place them at anyplace we want), we are more confident to tell where our picar is.

For example, I draw a picture of one image the camera captured when the picar is running as follow:



From the camera interface, we know that the size of each image is 640 x 480 (which means each row has 640 pixels and each column has 480 pixels). Besides, the results of decode function tells the pixels location of the four corners of the QR code. Afterwards, we can get the pixel location of the center of the QR code from the four corners. Suppose, the four corners location are (469, 132), (468, 232), (534, 134) (532, 230) respectively, the pixel location of the center is:

$$\left(\frac{469+468+534+532}{4}, \frac{132+232+134+230}{4}\right) = (501, 182)$$

The center of the image is at (320, 240), so the x-difference between the two centers is 181.

Based on this knowledge, we could tell the picar needs to turn left in order to let the QR code show at the center of image (closed-loop).

For convenience, I assume it is a linear relationship between the x-difference and the angle. (Angle = ka · x-difference)

Hw2 CSE276A Yunhai Han

In the other part, from the width and height, we could tell the approximate distance between the QR code and the picar.

For example, if we place the picar one meter in front of the QR code and the ~~reads~~ read outputs from decode function, it could tell us the width ~~of height~~ and the height of the QR code are 85, 86 respectively. And then, I place the picar 0.5 meter in front of the QR code and this time the width and the height of the QR code are 168, 173 respectively.

Hence, we could assume that there is a linear relationship between the ~~depth~~ distance and the scale of the QR code.

Based on the pixel coordinates and the scale, we could locate the ~~pixel~~ picar more accurately and effectively.

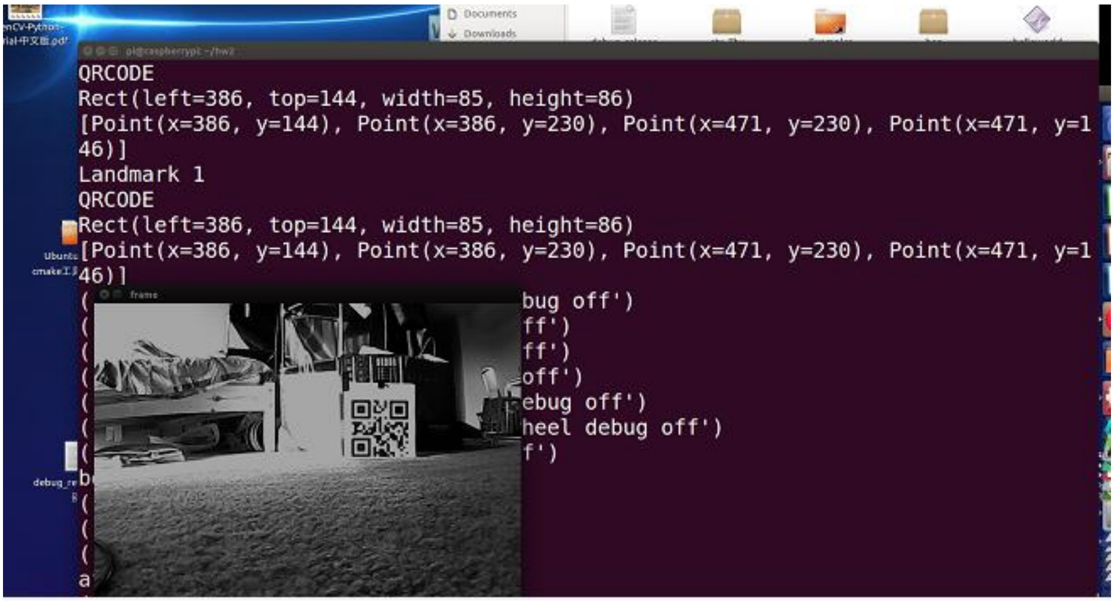
Moreover, we could also use the wheel odometry to record the position and orientation of the picar as in the homework 1. And when ~~the car~~ it is really near the target area, it could turn right and go to the next target area.

To sum up, from the visual information, we could determine the motion actions for the picar. With the wheel odometry, we could ~~record~~ update the location of the picar after each motion action has done.

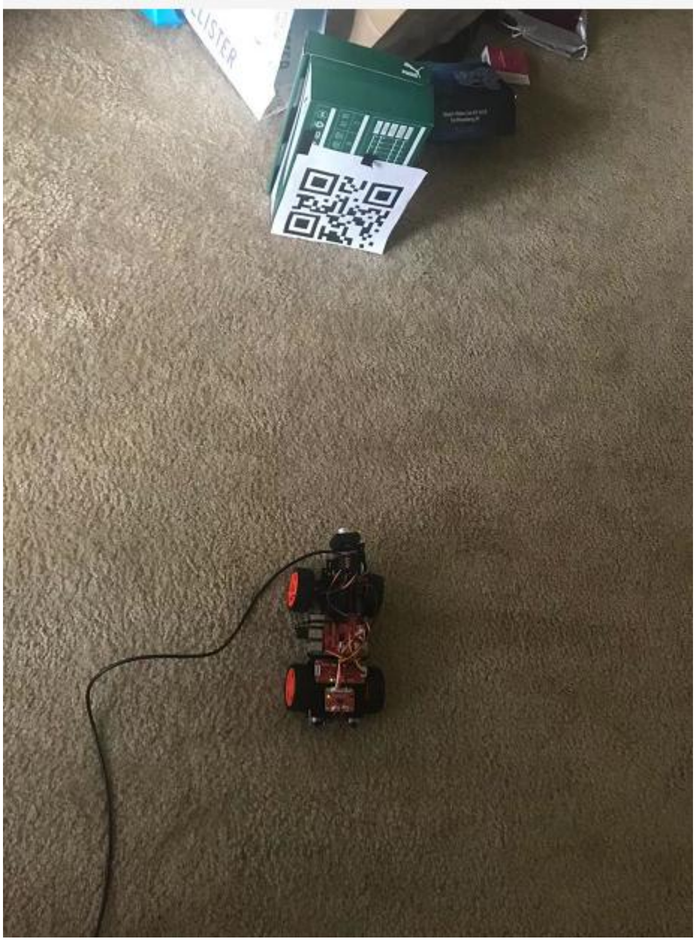
This above picture shows how I place the four QR codes. As what I say before, I use visual information and wheel odometry together. The visual information could determine the motion action of picar and the wheel odometry would update the pisition and orientation of picar each time after the action has done. When picar is very close to each QR code, it would turn right and find the next one.

As you see in the above picture, I place the four QR codes at the four corners of a square, so picar would finally get back to the origin.

In homework2, we could tell that the location of the QR codes are 100% accurate. However, in the real environment, nothing is with 100% centainty. In the later homeworks, we could introduce some other parameters to measure the uncertainty of the localization of the QR codes(landmarks) and the position of picar. Hence, the localization of QR could change based on the information where the camera find it and the position of picar could change either when the camera find a QR code. And I think, with the future improment, we could achieve a better result.



For these two pictures, from the first one, you can see that I place the picar one meter in front of the QR code and the second picture was the screenshot of the outputs of the codes(you can see that the width and height of QR code are 85,86)



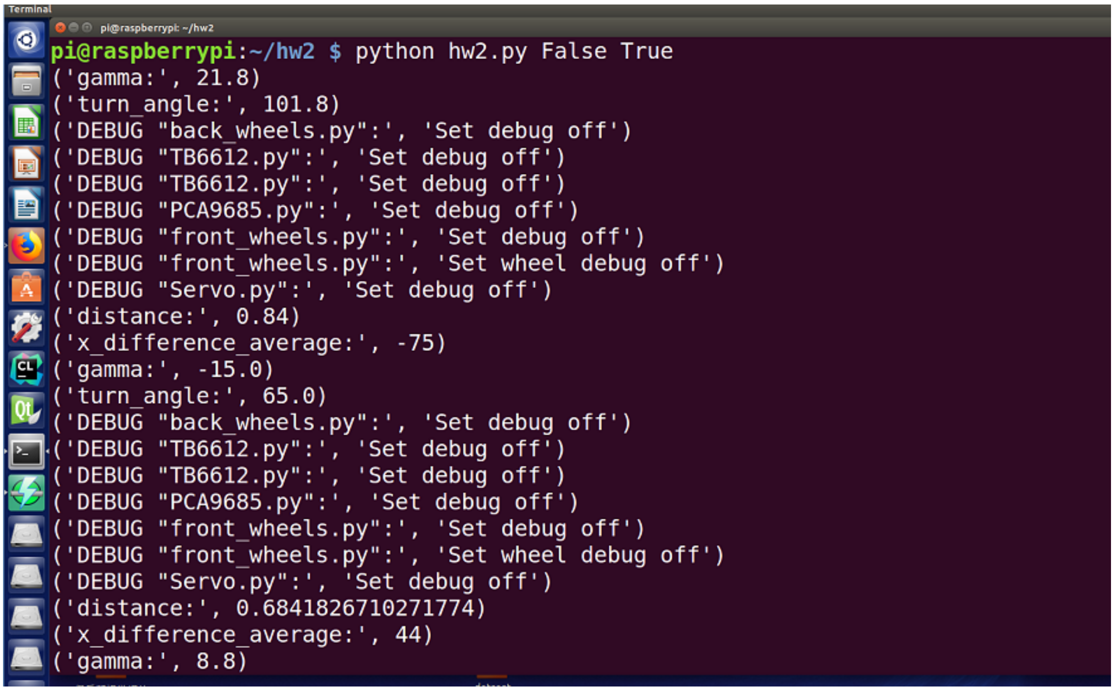
For these two pictures, from the first one, you can see that I place the picar 0.5 meter in front of the QR code and the second picture was the screenshot of the outputs of the codes(you can see that the width and height of QR code are 168,173)

Hence, I assume there is a linear relationship between the distance and the scale of QR code and this relationship could help me to localize our robot.

2.A description of the achieved results with comments on the accuracy of your solution.

The performance of the algorithm is illustrated in the video as picar has successfully passed all the positions.

Besides, I put a picture which shows the outputs of the algorithm.



```
pi@raspberrypi:~/hw2 $ python hw2.py False True
('gamma:', 21.8)
('turn_angle:', 101.8)
('DEBUG "back_wheels.py":', 'Set debug off')
('DEBUG "TB6612.py":', 'Set debug off')
('DEBUG "TB6612.py":', 'Set debug off')
('DEBUG "PCA9685.py":', 'Set debug off')
('DEBUG "front_wheels.py":', 'Set debug off')
('DEBUG "front_wheels.py":', 'Set wheel debug off')
('DEBUG "Servo.py":', 'Set debug off')
('distance:', 0.84)
('x_difference_average:', -75)
('gamma:', -15.0)
('turn_angle:', 65.0)
('DEBUG "back_wheels.py":', 'Set debug off')
('DEBUG "TB6612.py":', 'Set debug off')
('DEBUG "TB6612.py":', 'Set debug off')
('DEBUG "PCA9685.py":', 'Set debug off')
('DEBUG "front_wheels.py":', 'Set debug off')
('DEBUG "front_wheels.py":', 'Set wheel debug off')
('DEBUG "Servo.py":', 'Set debug off')
('distance:', 0.6841826710271774)
('x_difference_average:', 44)
('gamma:', 8.8)
```

gamma represents the steering wheel angle

turn_angle represents the value I set to the control function(front_wheel.turn())

(when the angle is equal to 80, picar would move forward straightly, so 101.8=80+21.8)

distance represents the distance beween picar and the target position

x_difference_average represents the difference between the image center and the center of QR code.

As you can see, the distance is decreasing as picar is approaching the target position, which you can see on the video.

Hence, it is proved that my algorithm successfully uses the visual information and achieves a better performance that it is in HW1 with open-loop control algorithm.

3.A short video that demonstrates the performance of your solution.

This is the Youtube link: https://youtu.be/XAa2mRP98v