New York University

Tandon School of Engineering

Department of Aerospace and Mechanical Engineering

ME-GY 6933

Advanced Mechatronics

Final Project

Cooperative robots based on OpenCV

Instructor: Prof. Vikram Kapila

Team member: Kota Tomaru, Jing Xia and Ziyu Wang

NYU_Tandon_Engineering_School_Advanced_Mechatronics_Final_Team_Project

Abstract

In this paper, a cooperative two-robot system is introduced. In this project, two robot cars can move an item to a specific position. OpenCV is applied for collecting information and making decision. At the same time, Arduino is used to relies motion control. The final goal is achieved.

Contents

1. Introduction

1.1 Background

Robots are widely used in modern society and will save more human source in the future. Robots can help human beings finish various tasks in different areas. Industrial area, manufactory robots help manufacturing in assembly line; in transportation area, autonomous cars transport goods and people from one place to another; in house caring area, human liked robots treat patients very well. However, all these missions. are achieved by one single robots. However, in the future, corporative robots will be more useful. Hiromasa *et al* paid attention on the corporate of two mobile robots applied in handling huge sized object **[1]**. Besides, B. H. Lee and C. S. G. Lee developed a trajectory plan method for avoiding collision of two robots in the same workplace **[2]**. Similarly, a map-based navigation method for two robots to work in the same workplace was developed by Min *et al* **[3]**. In this paper, computer version based method is applied. Open source computer vision (OpenCV) is a powerful image and video processing platform. Two robots with camera will make decision according to the image or video information.

1.2 Achievement

The final goal for the two robots is working together and moving one box to a specific position. This task can be divided into three progresses. In the first section, each robot moves towards to the box and grab one end of that box. In the second section, two robots will lift that box corporately. In detail, in this section, two robots will always achieve the end of box at different time. Therefore, the robot achieve box earlier will wait till the latter robot also achieve that box. In the final section, robots will lift the box up and move in the same direction. However, to move in the same direction, the actual movement for each robot is different. In detail, one robot should move forward and the other should move backward. In this way, the item can move in one direction. Therefore, robots use image information to decide to move forward or backward. Moreover, the specific position is decided by an ultrasonic sensor mounted on the rear end of each robot. In summary, information transformation represents the cooperation of two robots and the movement for each robot is related to computer version.

1.3 Hardware design and cost

For each robot, the hardware design flowchart is as following:

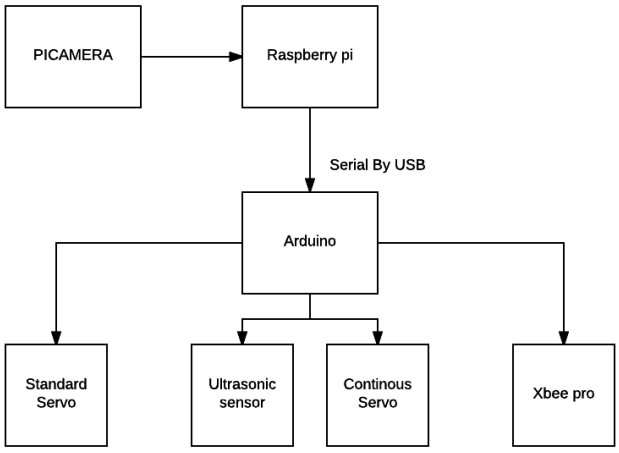

Figure1-3. Hardware Flowchart for each robot

From the structure, the Arduino controls the most parts of this project since Arduino has many open-resource libraries for using these parts. Generally, you set each function separately and combine them

together to make up your own function, which is more efficiency and easier compare to use GPIO pins to connect all the parts. For the raspberry, it could be thought as a master if the Arduino is a slave since many functions in this Arduino code is driven by the raspberry message sent by serial. The raspberry's task in this project is to read the image data and process it, then send corresponding cases' command to Arduino.

Table 1. Cost of hardware in this project

Name	Quantity	Unit Price
Boe-Bot Gripper Kit with Arduino motor shield	2	$59.99
Arduino Uno R3	2	$20.99
Raspberry pi 3 Model B with 16G SD CARD	2	$35
PiCamera	2	$25
Ultrasonic sensor	4	$29.99
Xbee Pro	2	$28.50
PowerBank battery	2	$13.99
10 AA Duracell battery PACK	2	$10
		Total: $509.96

For the Boe-bot Robot, it contains two parallax continuous servos for two wheels and standard motor for gripper. The reason that each robot equipped with 2 ultrasonic sensors is that the front one could make robot detect object and the rear one could make robot avoid obstacles when two robot carrying object together.

For the pi-camera, the v2 Camera Module has a Sony IMX219 8-megapixel sensor. The size of it is pretty fit for this robot, which could be easily mounted on the front of the robot.

For the raspberry pi 3 model b, it has a 1.2GHZ 64-bit quad-core ARMv8 CPU, which could handle the computation of OPENCV. And it provides 4 USB port, the Arduino could be connected to raspberry pi by using USB cable.

In this project, each robot have three layers, the first layer is occupied by a metal bar of gripper which is connected to standard servo motor and front gripper. The second layer is the place which put the Arduino board and Xbee Pro module. Raspberry pi is put the on the top layer with camera. Also, the powerbank that supports raspberry pi is put the on the rear of third layer.

1.4 Software

For the raspberry pi 3 model b, the system installed on it is Raspian Jessie, which is a Debian-based operating system. For finishing this project, there are some software that must be installed as following:

1. Python 2.7 or Python 3
2. Open cv 3.1.0 or 3.2.0
3. VNCserver
4. Arduino IDE Arm LINUX edition

2. Theory

2.1 Arduino part

In our project, Arduino is in charge of controlling mobile robot itself to carry the object. For the sake of carrying the objects, Arduino has several steps.

2.1.1 Receiving desired direction information

The Arduino receives information of desired direction from Raspberry pi 3 before approaching to the objects. As Figure 2-2-1 shows, Arduino and Raspberry pi3 are connected by serial connection. After the raspberry pi 3 detects image, Arduino receives information as integer numbers. As a result, both robots can know desired direction for which each robot will have to carry the objects as shown Figure 2-2-2.

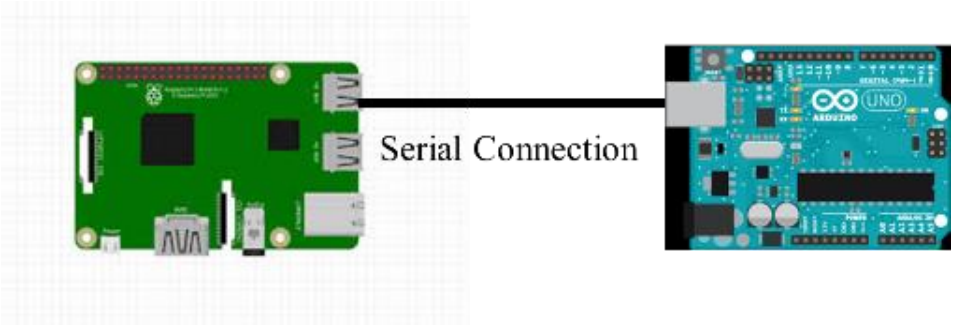

Figure 2-1-1: Connection between Raspberry Pi and Arduino

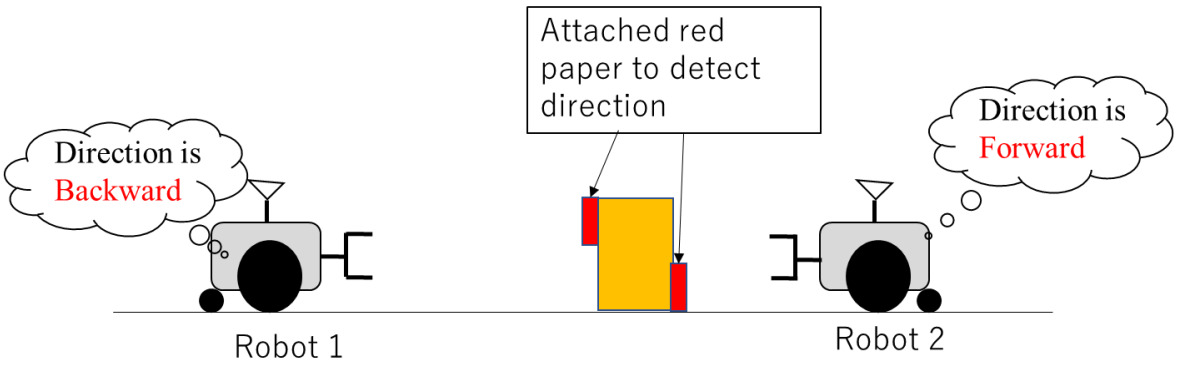

Figure 2-1-2: Different working principle for two robot cars

2.1.2 Approaching to the objects

After knowing desired direction to carry the objects, each robot starts to approach to the objects. At that time, each robot uses a ultra sonic sensor attached to the front side to measure distance between the robot and the objects. When the measured distance is under criteria (= 5cm), the robot will stop and start to grasp.

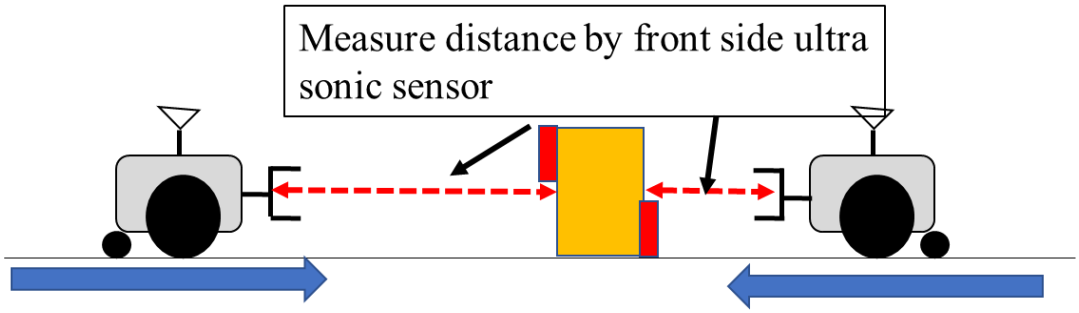

Figure:2-1-3: Use Ultrasonic sensor to make decision

2.1.3 Grasp the objects and start Xbee communication

After grasping the objects, each robot will start Xbee communication to inform each desired direction to carry to the other robot. For example, as Figure 2-2-4 shows, when robot1's desired direction is "backword" and robot2's desired direction is "frontward", robot1 will send message "R" and robots will send message "F". Moreover, by receiving Xbee message from the other robot, each robot can know other side robot also finishes approaching and grasping. Then, until when each robot receive message "F" or "R", both robots will not start to carry the objects.

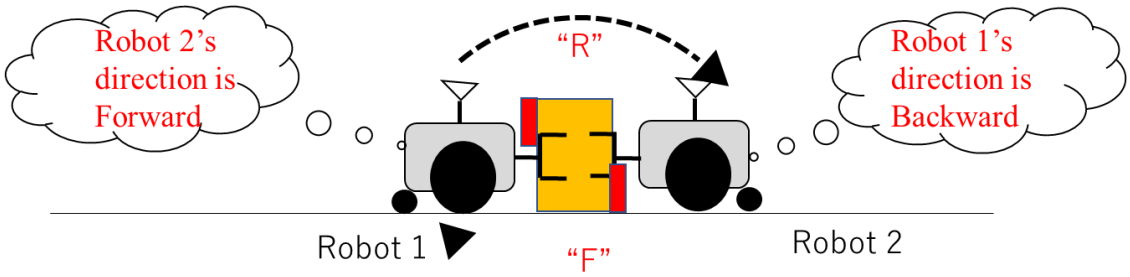

Figure:2-1-4: Different movement of two robots

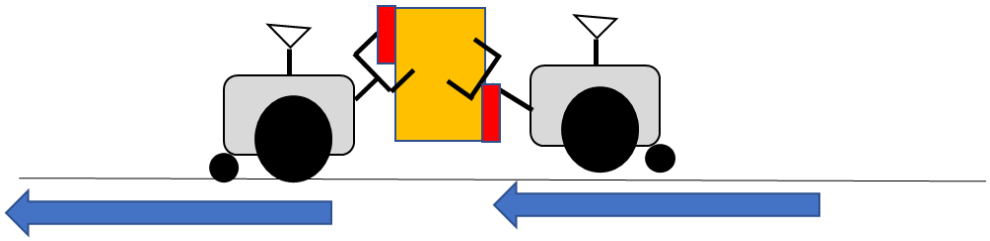

Figure:2-1-5: Effect of two robots work together

2.1.4 Carrying the objects to the desired direction

While both robots are carrying the objects, only front robot measure distance between the robot and obstacles by ultra sonic sensor attached to the rear side. When the measured distance is less than 5cm, the front side robot will send message "S" through Xbee communication. After sending the message "S", the

front side robot will stop and release the objects. For the other side robot, after receiving message “S” from the front side robot, the robot will stop and release the objects.

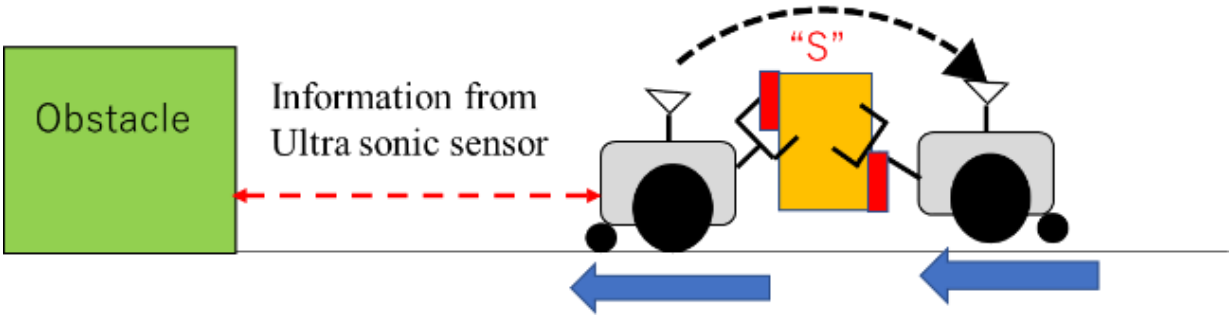

Figure:2-1-6: Use Ultrasonic sensor to stop movement

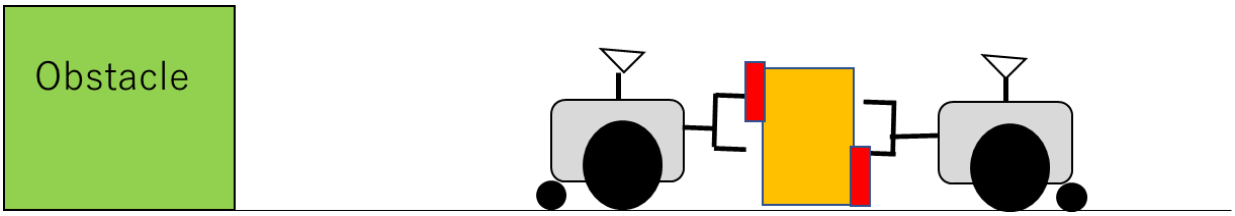

Figure:2-1-7: Movement stop

2.2 OpenCV part

For the Python code, there are some libraries which required to installed on the python as following **[5]**:

1. Picamera
2. CV2
3. Serial
4. Imutils

Picamera library contains all the supporting files provides a pure Python interface to the raspberry pi camera module.CV2 is the library that OPENCV library on the python, which make users can use function in the OPENCV to process the image.

Serial This module encapsulates the access for the serial port. In this project, the serial port is from raspberry pi to the Arduino. In the code, you must declare Such as

```
Arduino.write= serial.Serial('/dev/ttyACM0',9600
```

For ttyACM0, this is the port name that Arduino connects to the raspberry pi by usb, the 9600 means that hat the serial port is capable of transferring a maximum of 9600 bits per second.

Based on the python software foundation website, Imutils contains basic image processing functions such as translation, rotation, resizing, skeletonization, displaying Matplotlib images, sorting contours, detecting edges, and much easier with OpenCV and both Python 2.7 and Python 3. In the beginning, the first idea to approach this object is to process the video steaming. However, it will be a huge time delay if use video streaming based on the some other group's experiences。

For the code part, first it requires to set up an range for lower and upper of the object that camera need to detect. The format to define this range is RGB value. The difference of lower and upper boundaries is (RH-RL)*65536+(GH-GL)*65536+(BH-BL).

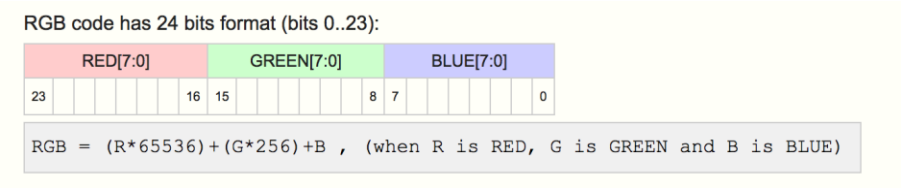

Figure 2-2-1. RGB value has 24bits

The camera detecting function will be influenced by light a lot so that the boundaries set is large in this project. The color should be detected is red. However, it can also detect purple. A suggestion to make more easily to detect color is using white paper as background to do it. Then the code will translate the RGB value to HSV value that would more decrease the influence of the light. After this, the image will be blurred to make the image smooth. Finally, the middle of red in the image can be obtained. The positions of the center of mass are defined with different meaning. Figure shows the flow chart of all criteria.

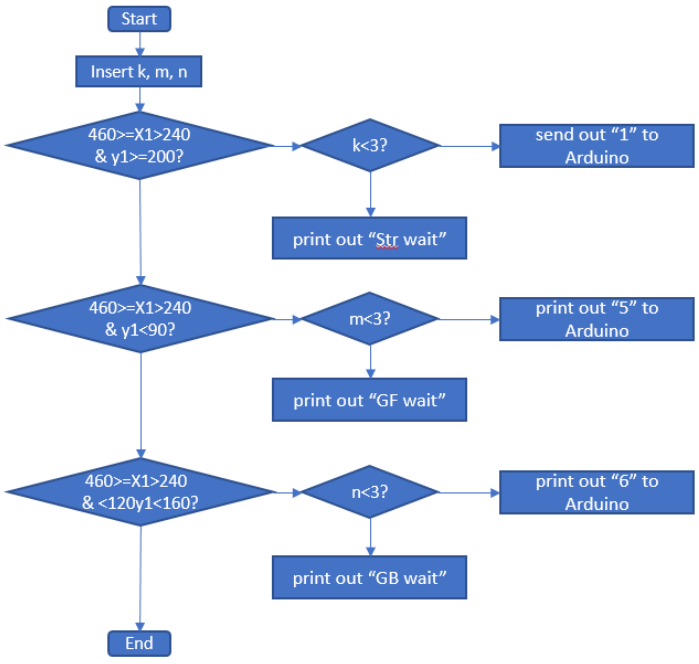

Figure 2-2-2: Flow chart of different cases

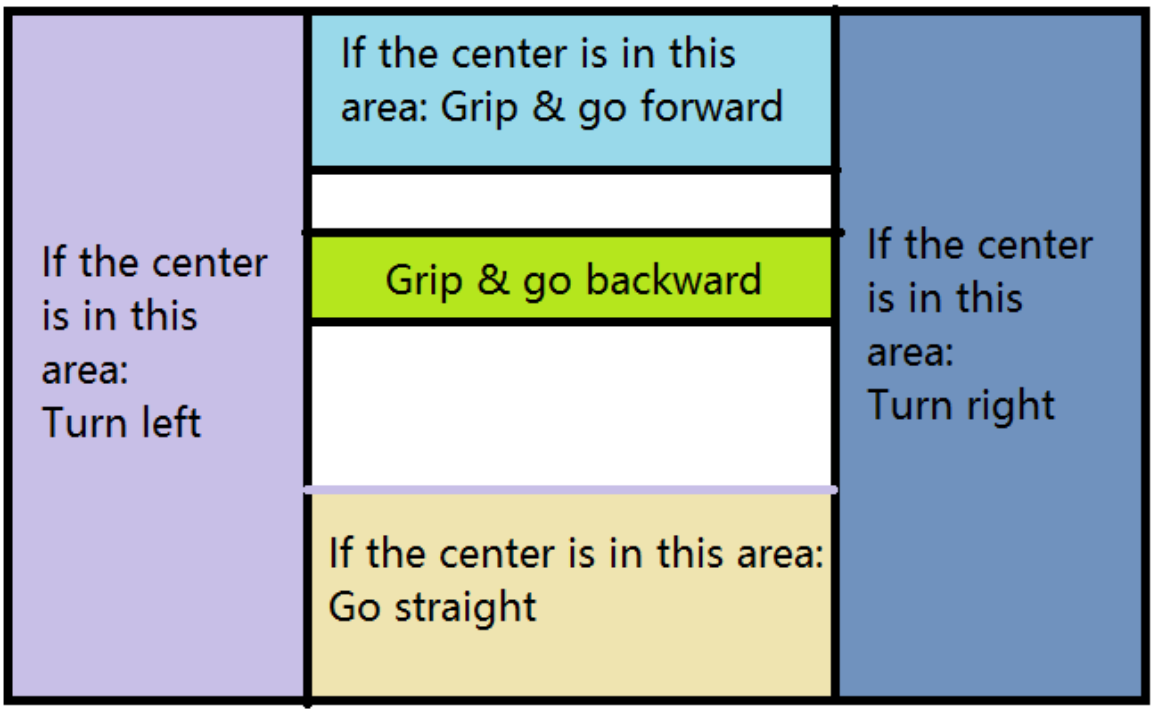

Figure:2-2-3: Different criteria for decision making

3. Results

As mentioned in introduction part, the final goal for two robots is to move an item to a specific position. This goal is achieved by the end of the project. Two robot car moved to and gripped the box and then send the box to specific position.

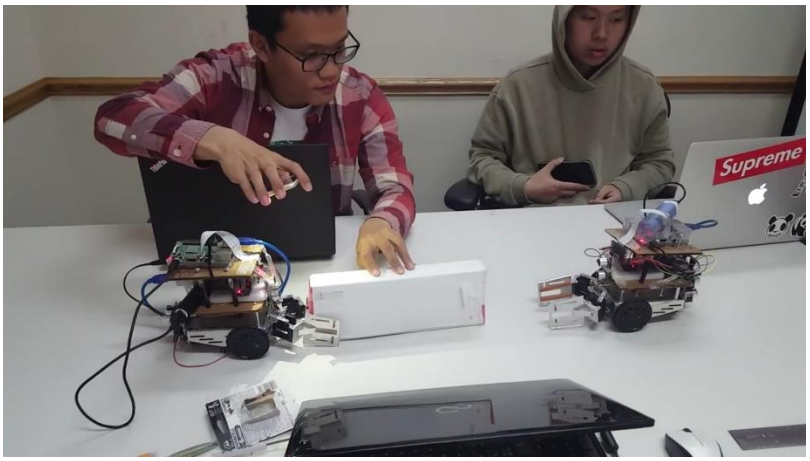

Figure 3-1: Robots grip item

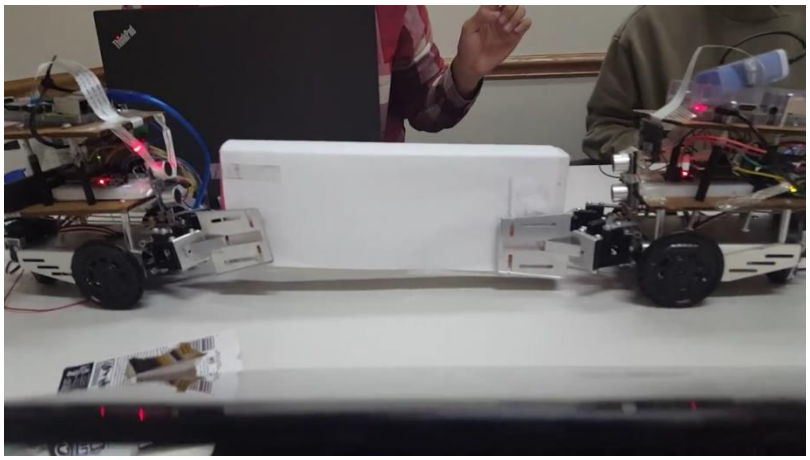

Figure 3-2: Robots lift item up

Two directions were tested. In the first test, robot 1 faced to the mark which indicated moving forward and robot 2 faced to the mark which indicated moving backward. In the second test, robot 1 faced to the mark which indicated moving backward and robot 2 faced to the mark which indicated moving forward. Figure 3-3(a), 3-3(b) and 3-3(c) show the object and marks for different movement direction. Because the camera mounted on robot cars are upside-down, the marks in screen is slightly different from real mark.

Figure 3-3(a): Overview of item Figure 3-3(b): Mark for backwards Figure 3-3(c): Mark for forwards

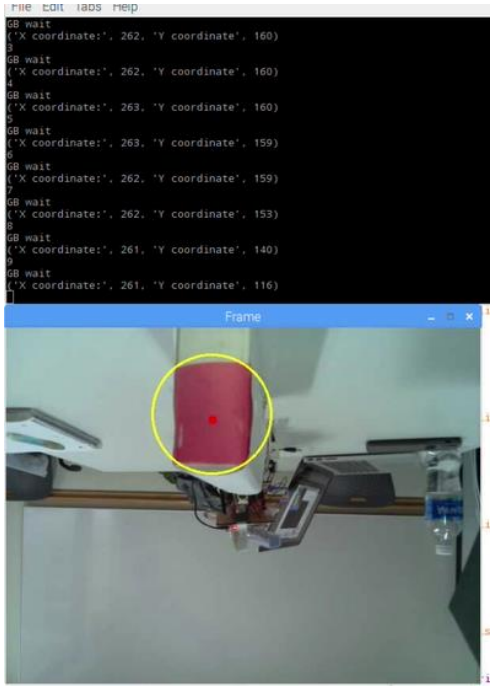

Figure 3-4: Results shown on Raspberry pi when robot move backwards after grip

Figure 3-5: Results shown on Raspberry pi when robot move forward after grip

4. Discussion

4.1 Reasons for failure

4.1.1 Lighting environment

During testing progress, lighting condition cost huge influence to the testing result. The detection of red color mark was unstable in different lighting environment. To solve this problem, extra lighting source was applied during the testing.

4.1.2 Internet connection

In this project, two robots were involved. Thus, to control two robot through Raspberry Pi need two computer and two screens. However, because of the limited equipment, two VNC viewer were used. This cost huge mass. Because of poor internet connection, VNC viewer could offline several time during a test. This problem wasted huge amount of time and influenced the accuracy of output image information.

4.2 Improvement in the future

In this testing robot, color tag is still not a good choice for detecting if the light condition is not good. The better choice should be use AR or April tag since it can represent different message from its pattern.

From mechanical part, the gripper is only one DOF, which has a limitation to seize some types of object. It is possible to put an 6 DOF manipulator on the robot if the robot size could be increased like Romba. And the driven system now is two wheels using differential which is not holonomic. In this case, they can only move on a certain line when these two robots grab one object together. For easy control consideration, the best choice will be mecanum or omni wheels.

The number of robots to control is also could be increased since that it is convenient to use ROS to control many robot as swarm to do some certain task. Users can send command on hub, the robot could do same task by subscribe same topic.

5. Conclusion

This project proposed the method that two mobile robots can carry an object with each other to desired direction based on the image information. This method can make robots collaborate flexibility using environment information. This project achieves its goal of providing a better productivity in a lot of places by using image detecting method to adapt environment. In the future, the robots should go through different routes to approach to objects. Then, the robots should move exactly following desired trajectories. To achieve this, for example, encoders can be to detect actual movements of robots apply feedback control.

Reference

[1] H. Kamogawa, Z. Liu, and J. Ota, "Handling of a large irregularly shaped object by two mobile robots," ***2011 IEEE International Conference on Robotics and Biomimetics*****, 2011.**

[2] B. Lee and C. G. Lee, "collision-Free Motion Planning of Two Robots", ***IEEE Transaction on System, Man and Cybernetics*****, vol. 17, no. 1, pp. 21-32, 1987**

[3] M.-G. Lai, W.-T. Zeng, and C.-F. Juang, "Navigation for two fuzzy controlled cooperative object-carrying robots in concave maps with the consideration of dead-cycle problem," ***2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*****, 2016.**

[4] A. Rosebrock, "Ball tracking with OpenCV-PylmageSearch", ***PyImageSearch*****, 2017. [Online]. Available: http://www.pyimagesearch.com/2015/09/14/ball-tracking-with-opencv/ [Accessed:13-May-2017]**