Course No.: ME-GY 996X

An Autonomous Mobile Robotic System to Assist Disabled Persons in Daily Activities

M. S. Project Report Submitted to the Department of Mechanical and Aerospace Engineering, Tandon School of Engineering, New York University, Brooklyn, NY

Submitted By

Haiming Gang

N10533017

May 15, 2017

Abstract

This report presents the development of a mobile robotic system combined with a manipulator, image processing and navigation with mobile devices to assist disabled human in an indoor environment. The project utilizes a Microsoft Kinect, three microcontrollers, a mobile phone, a mobile robot base and an arm robot. Human user chooses different objects to be delivered by the robot for the human through mobile phone interface, and the device can drive the manipulator to the preset picking-up according to known map and SLAM technique, recognize the targeted object and measure the distance between the object and the manipulator in the video frame got from the Kinect according to different object classifiers. After image processing, the mobile phone sends the position information to the raspberry pi, and the microcontroller utilizes the inverse kinematic equation to calculate angle of each joint in the robotic arm, and finally sends this data to the Arbotix-M for transforming the angular positional information to electrical signal and controlling the action of the robot. After picking up, the robot sends the object to a preset position around the user, and the user can give the robot a command to release the object. The prototype was developed that was successful in self-navigation, recognizing various objects as well as in picking and handing over the object based on the command of the user through the smartphone interface. An evaluation scheme has been proposed to evaluate the usability of the system.

Keywords: Autonomous mobile robot, SLAM, robot manipulator, object recognition, image processing, assistant robot, disability, daily activity, user interface

1 Introduction

In this day and age, human is used to the convenience of technology for saving their time. For instance, robot comes into the life of humans, and helps humans perform various tasks. Human always excepts robot becomes smarter and cooperative and can deal more varied tasks. This needs a smarter robot wherever possible, which forms the objective of this project. Human always expects a robot that can work as an intelligent assistant that takes care of all domestic repair and maintenance tasks. The smart robot can help humans pick and place some objects when they are focusing on their own affairs, and this function can be achieved by the command from humans and even from the robot itself when it has artificial intelligence.

Nowadays, the most of the robots work following commands that humans set in the machine or send to the manipulators, especially the industrial robotic systems usually lack simple user interfaces. In addition, some of the robots need supervision, but the real condition is that human cannot spend lots of time to look through the robot, so they want a robot that can finish some task by itself. The question is, how can the operations of a robot be made much easier for humans? Hence, the operation of robot needs to be easier for the users. However, such easy operation methods are still not enormously available.

This is why, we need the robotic system that can complete some tasks with easy user operation and includes some artificial intelligence. The smart robot should combine a few important functions that make it convenient to use. The arm robot may utilize a Microsoft Kinect Xbox 360 camera to implement SLAM and obtain position information of the object. With the popularity of mobile phones, humans spend lots of time on mobile phones, human becomes more likely to use mobile phone to finish their daily works, and the processor of smart phone may also provide stronger processing ability than that provided by the microcontroller. Furthermore, a mobile phone can be easily implemented remotely when user is away from the arm robot. In the project, the system consists of two main parts, one is a mobile base, and another is a manipulator. The mobile base can drive the manipulator to the pick-up point, and

then the manipulator grasps the object, and finally carries the object to the user. A few objects placed within the workspace of the robot arm are recognized by the robot through image processing using the images frame taken by the Kinect. The human user can command the robot to manipulate a particular object. The whole process is divided into several steps:

i. The human will connect the mobile phone with robot, and then choose the object by clicking the corresponding button.

ii. The robot will move to the preset pick-up point according to different the object.

iii. The robot recognizes the object and obtains the position information

iv. The manipulator picks up the object

v. Robot delivers the object to the user

The goal of this project is developing a robotic system that can help human complete some picking and placing tasks in their daily life. This device adapts various environments which contain many irrelevant objects with multiple colors and is portable. The robot can complete the navigation automatically based on the known map. The robot can organize several different objects according to human intent, so that human can use the manipulator remotely and unsupervised. For normal human, the manipulator can help them pick up objects when they have some other tasks and control the robot to complete some tasks remotely or set in advance. For disabled persons (patients) who have no or limited mobility, the robotic system can assist the persons in their daily activities.

This report is organized in the following ways: Section 2 presents the system components and introduces how the system works; Section 3 presents the experiment and results for system evaluation; Section 4 presents acknowledgment; Section 5 presents the conclusion and expectation for this project.

2 Development of the System

2.1 The Overall system

The goal of project is designing a robotic system which contains a mobile robot base and a manipulator. This system can be operated remotely through a smart phone, the system can complete autonomous navigation of a known map, can recognize objects based on image processing, pick up the recognized objects and carry the objects to the users at designated locations automatically. The system consists of a mobile phone/computer, a raspberry pi 3, a raspberry pi 2, a Microsoft Kinect Xbox 360, an iRobot create, an Arbotix-M robot controller and a 5-DOF manipulator which contains shoulder pan, shoulder lift, elbow joint, wrist joint and finger joint. Fig.1 shows the overall robotic system.

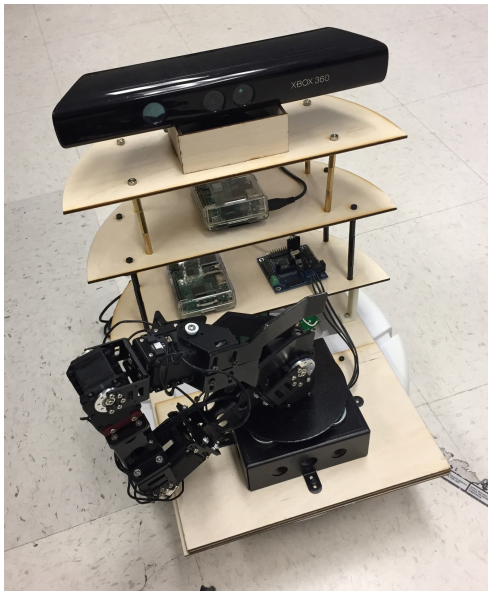


Figure 1: Robot system

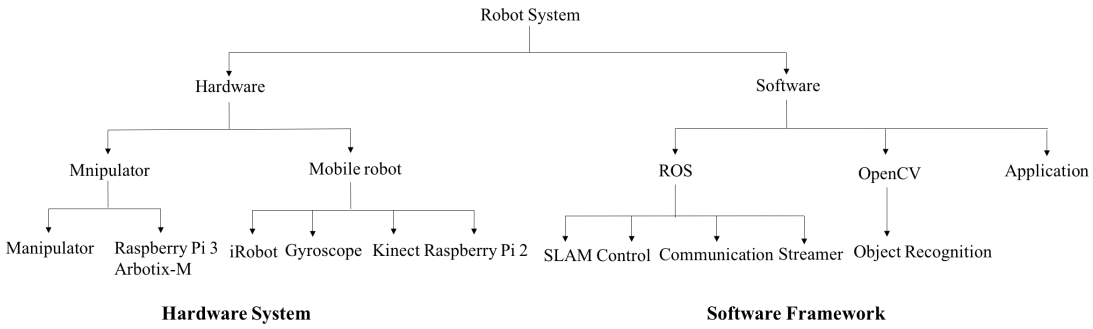


Figure 2: The configuration of Robot System

As Fig.2 shows, the mobile robotic system contains 1 iRobot create, 1 manipulator with 5 DOF, 1 Microsoft Kinect Xbox 360, 2 raspberry pi (raspberry pi 3 controls the manipulator and raspberry pi 2 controls the mobile robot base), 1

arbotix-M microcontroller and 1 battery bank with 10000mAh. The whole process would be as shown in Fig. 3:

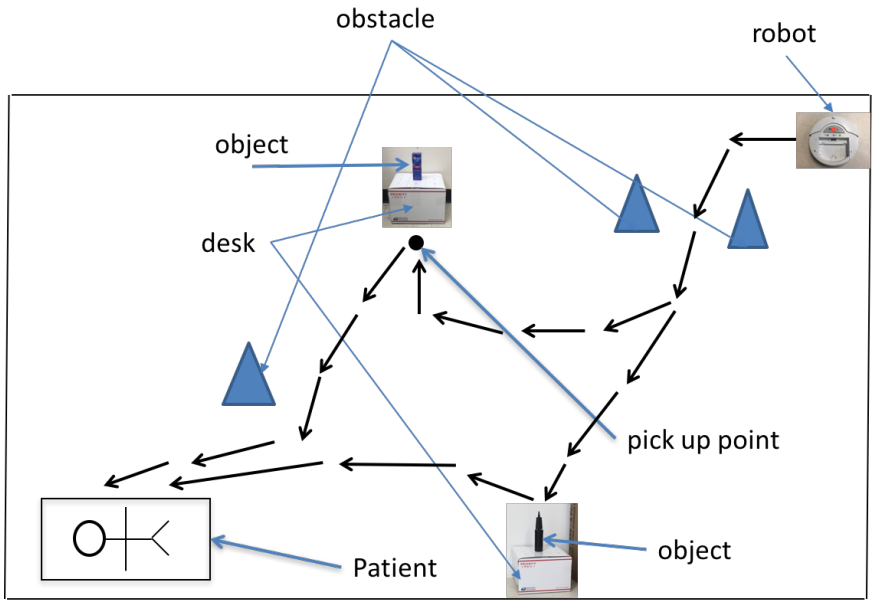


Figure 3: The whole working process

The process of the whole system works as follows:

i. The user opens the App installed in the mobile device, and clicks “connect” button to connect the mobile device with the robot, and then chooses the object which he/she wants to pick up, and finally clicks the “go” button to trigger the robot.

ii. When robot receives the command, it begins path planning from the current position to pick-up position which is pre-stored in the program.

iii. The robot utilizes the SLAM technique to navigate itself to the goal position.

iv. When robot arrives the pick-up point, the robot adjusts itself to a suitable position with respect to the object within the workspace of the manipulator.

v. The Kinect sends back the RGB image to the mobile device through mjpg-streamer, and the mobile device analyzes the image frame and obtains the pixel position information, and finally sends it to the robot.

vi. The Kinect detects the depth between the sensor and the object according to pixel position provided by previous process, and transforms the distance to the distance between manipulator and object.

vii. The manipulator picks up the object according to the position information.

viii. The robot begins path planning from pick-up point to the position of user which is pre-stored in the program, and navigates itself to this position.

ix. The user clicks “release” button to open the fingers of the manipulator and then catches the object.

2.2 The mobile part

2.2.1 The iRobot Create

In this project, we choose iRobot create as a mobile platform as shown in Fig.4.

Figure 4: iRobot create

2.2.2 The Microsoft Kinect Xbox 360

The specifications of the Microsoft Kinect Xbox 360 is given below and the picture of the Microsoft Kinect Xbox 360 is shown in Fig.5.

Viewing angle: Field of View (FoV): 43° vertical x 57° horizontal; Vertical tilt range: ±27°; Frame rate (color stream): 640x480@30 frames per second (FPS); Frame rate (color stream): 320x240@30 frames per second (FPS); Max Depth Distance: 4.5m; Min Depth Distance: 0.5m.

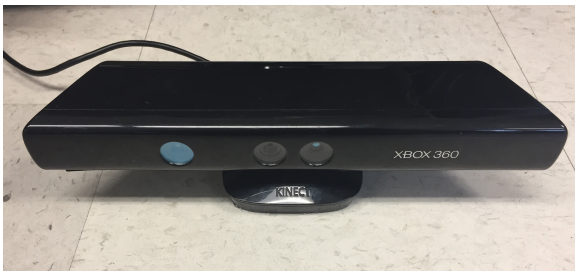


Figure 5: Microsoft Kinect XBOX 360

2.2.3 Raspberry pi 2 Model B

Raspberry pi 2 Model B is shown in Fig.6.

A 900MHz quad-core ARM Cortex-A7 CPU; 1GB RAM; 4 USB ports; 40 GPIO pins; Full HDMI port; Ethernet port; Combined 3.5mm audio jack and composite video; Camera interface (CSI); Display interface (DSI); Micro SD card slot; VideoCore IV 3D graphics core

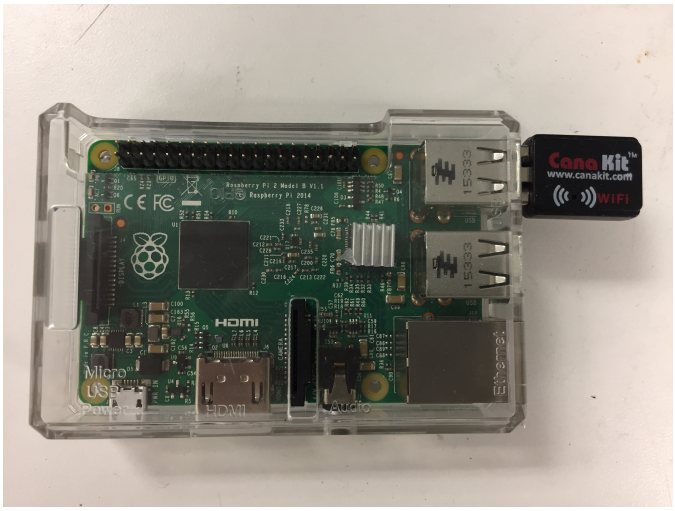


Figure 6: Raspberry pi 2 Model B

2.2.4 Mapping and Navigation

With the aim of traveling toward the objects and delivering it, the robot will have to travel according to a generated path. To do so, mapping techniques are required.

In this system, the robot utilizes ROS packages. In particular, “gmapping”[1] will have to be implemented in order to extract the data from Kinect while mapping. For better mapping and navigation, the robot uses external ADXRS610 gyroscope with a range ±300°/sec. The Kinect is able to produce laser data as well as RGB and depth images that will help in enhancing the mapping process. The ROS parameters were also adjusted accordingly with respect to the position of the Kinect. This

package contains a ROS wrapper for OpenSlam's Gmapping [2]. The gmapping package provides laser-based SLAM (Simultaneous Localization and Mapping), as a ROS node called slam_gmapping. Using slam_gmapping, the robot can create a 2-D occupancy grid map from laser and pose data collected by a mobile robot. The map built by gmmaping is shown in Fig.7.

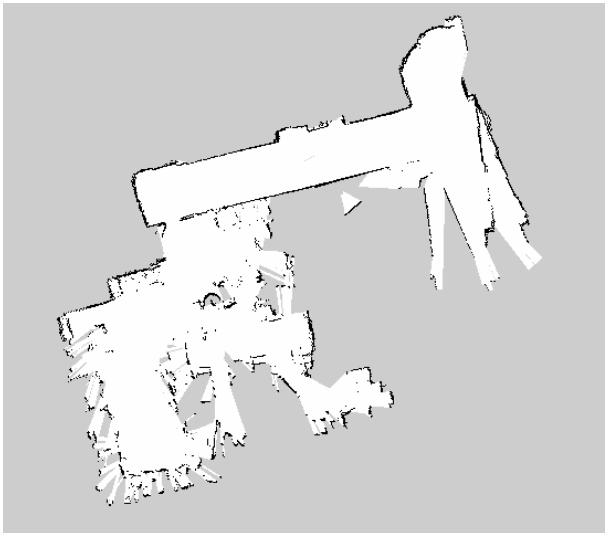


Figure 7: The map built by gmmaping

The robot is able to navigate automatically based on the known map built previously. The ROS package mainly utilizes amcl [3][4] which takes in a laser-based map, laser scans, and transform messages, and outputs pose estimates. On startup, amcl initializes its particle filter according to the parameters provided. The package also integrates the path planning and obstacle avoidance based on real-time laser data provided by Kinect and position estimation according to odometry and gyroscope. In other words, the robot plans a shortest and smoothly path from the initial position to the goal position, and this path will adjust according to real-time laser data, for example, if the Kinect finds that there is an obstacle on the planned path, the robot will search a new path to avoid the obstacle based on the known map and new environment information. The robot cannot arrive preset picking-up point accurately, because amcl is a probabilistic localization system for a robot moving in 2D and there is linear and angular error for odometry and gyroscope, and the real position where robot arrives actually is a random distribution around the preset picking-up point, in addition, the minimum measurement distance for Kinect is 50cm and the valid range for manipulator is 35~46cm. The robot needs to tune the position so that it can pick up object properly, as the process shows in below. Logic flow of

pick-up process is shown in Fig.8. There are two places which needs to pay attention to it: The depth is the distance between object and manipulator; if the distance between object and Kinect is less than 50cm, the depth will be considered as 0cm.

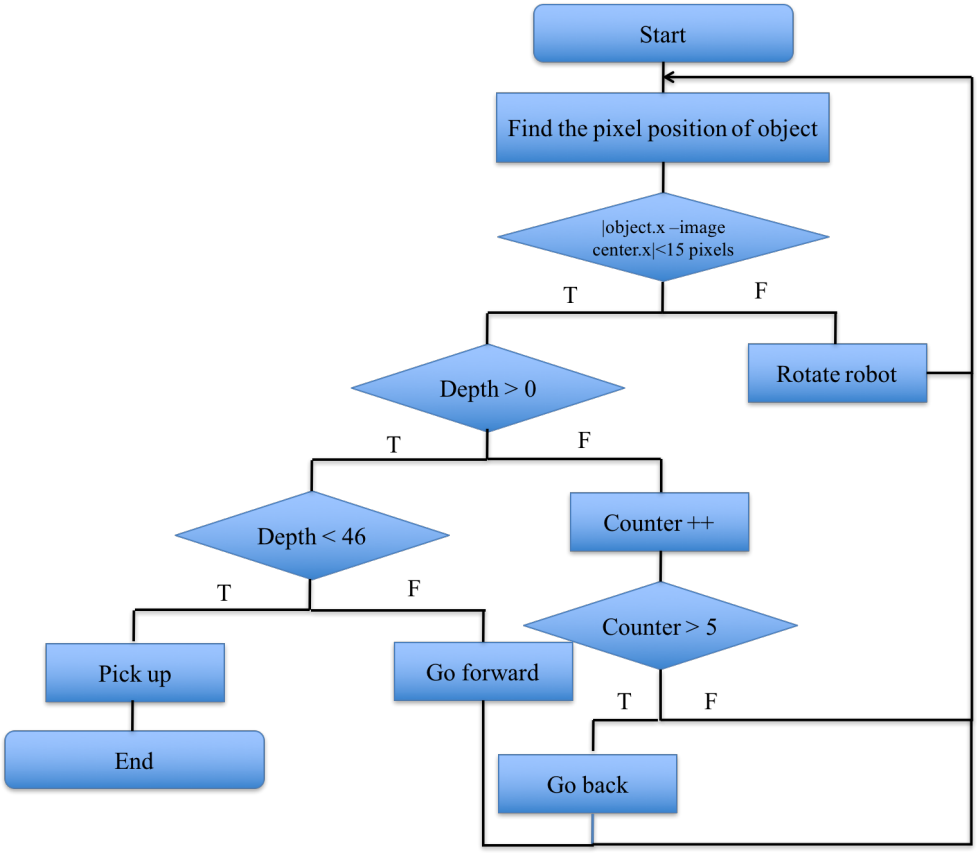


Figure 8: Logic flow of pick-up process

2.3 The Mobile Phone

We used a mobile phone. The specifications of the mobile phone are given below:

iPhone 5; ios 10; Apple A6 chip;1.3 GHz dual core 32-bit ARMv7-A CPU

2.4 Manipulator part

2..4.1 Raspberry pi 3 Model B

Raspberry pi 3 Model B is shown in Fig.9. A 1.2GHz 64-bit quad-core ARMv8 CPU; 802.11n Wireless LAN; 1GB RAM; 4 USB ports; 40 GPIO pins; Full HDMI port; Ethernet port; Combined 3.5mm audio jack and composite video; Camera interface (CSI); Display interface (DSI); Micro SD card slot (now push-pull rather than push-push); VideoCore IV 3D graphics core.

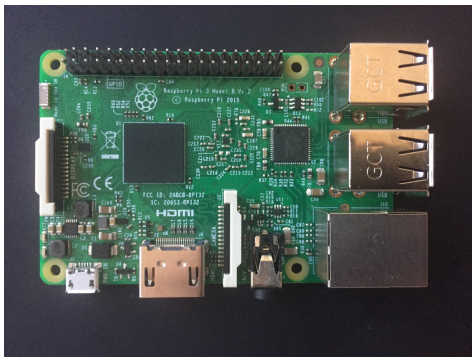


Figure 9: Raspberry pi 3 Model B

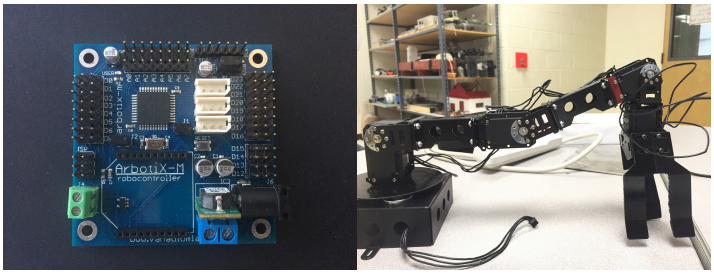
2.4.2 The ArbotiX-M

The specifications of the Arbotix-M are given below and the device is shown in Fig 10(a),

16MHz AVR microcontroller ; 2 serial ports, 1 dedicated to Bioloid servo controller, the other to the XBEE radio/FTDI programming; 3 TTL DYNAMIXEL 3-pin style ports onboard, plug TTL AX/MX DYNAMIXEL servos directly in; 28 Digital I/O, 8 of which can also function as analog inputs; Servo style 3-pin headers (gnd, vcc, signal) on all 28 I/O pins.

2.4.3 The Robot Arm

This manipulator is shown in Fig 10(b) that has 5 degree of freedom, and we use 2 MX-106, 1 MX-64, 1 MX-28 and 1 AX-12A Dynamixel servo motor.



(a) (b)

Figure 10: (a)An Arbotix-M microcontroller; (b) The Robot Arm

Figure 11: Interface in mobile phone

Human clicks the "connect" button on the interface shown in the Fig 11 to connect the phone with the robot, and then clicks "go" button to ask the robot to go to the pick-up point, and finally chooses the object by clicking the corresponding button on the touch interface. After receiving the "go" command, the robot begins to move to the pick-up point based on the map stored in the microcontroller.

The navigation is mainly based on turtlebot_navigation package in ROS installed in the microcontroller. This package utilizes Adaptive Monte Carlo Localization for localization task, and this package also contains path planning which uses Dijkstra's algorithm and collision detection function. After arriving pick-up point, the robot adjusts itself to find a suitable position for picking up. There is tolerance for the localization of the robot and the valid distance of Kinect is 50cm which means that the minimum measurement distance for the Kinect can provide the position information of the object for picking task is 50cm.

The robot begins picking up object after adjusting the picking-up position, and the Kinect sends the RGB image to the ios device through mjpg-streamer from raspberry pi which is connected with the Kinect. The ios device implements the image process after receiving the frame from the Kinect based on Haar casscase classifier stored in the ios device. The program recognizes object by classifier trained in advance and records the pixel position of the bottom part (object), and then using the pixel position finds the corresponding distance in depth image provided by the Kinect and calculates the orientation of the object between the center of the image and the position of the bottom of the object. Next step is sending the position and orientation

information back to raspberry pi, and then calculating the real distance between the manipulator and the object.

Raspberry pi calculates the joint position of each joint by inverse kinematic [5] algorithm after receiving the relative position data, and then sends the rotational angle of each joint to Arbotix-M. Arbotix-M coverts the angle to electrical signals, and then sends signals to the arm robot for picking up an object. After picking up, the robot goes to the goal position setting in the code, and the goal position is close to the patient(subject) so that the patient can easily receive the object. In this process, the robot also utilizes the navigation package working on ROS platform. Finally, the robot release the object after arriving at goal position and the patient can get the object. The robot can repeat the same process if the patient needs other objects.

2.5 Haar cascade classifier [6]

Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this reason, the haar features approach shown in Figure 12 are used. They are just like convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle [7].

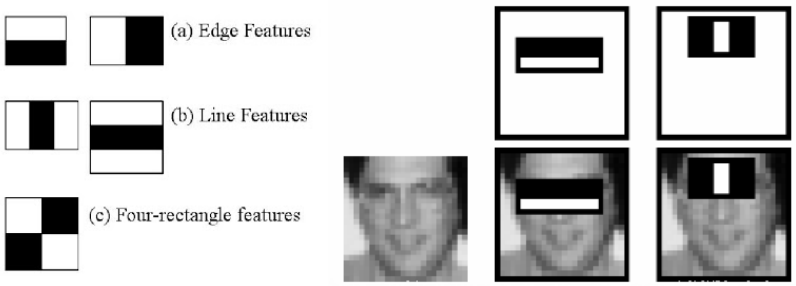


Figure 12: Haar feature approach

For example, consider the images shown in above. The top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. The haar feature reflects the change of the gray scalar of the image.

For the training process, we do grouping the features into different stages of classifiers and apply one-by-one (normally first few stages will contain very less

number of features). If a window fails the first stage, we discard it. We don't consider remaining features on it. If it passes, we apply the second stage of the features and continue the process. The window that passes all stages is an object region.

In this project, five objects are considered as the target objects shown in Fig.13. For the purpose of training faster in training phase and running program faster in image processing phase, the small size classifier is better than the large size classifier, although the accuracy of reorganization can decrease. In the training process: 700 positive images and 3500 negative images for the box classifier; 550 positive images and 2750 negative images for the beer classifier; 700 positive images and 3500 negative images for the toothpaste box classifier; 600 positive images and 3000 negative images for the pump classifier; 450 positive images and 2250 negative images for the cup classifier are considered.

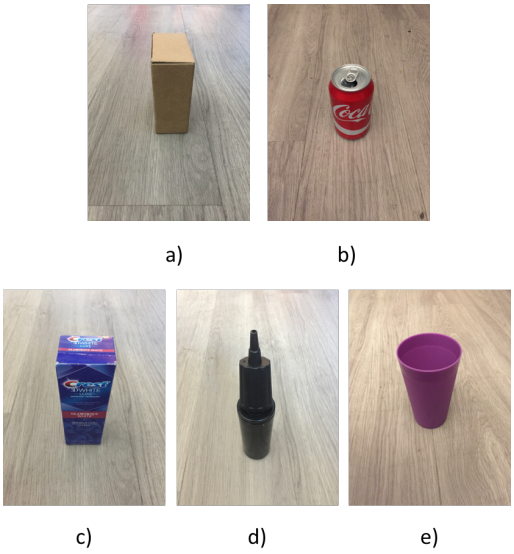


Figure 13: a) Regular box, b) Cola can, c) Toothpaste box, d) Pump, e) Cup

For better distinction of similar objects after recognition process, we use different BGR value shown in Fig.14 to assist in distinguishing process. According to sampling for each object, we find that the values of BGR are slightly different, and we use this difference to design restriction condition for noise elimination so that the system can obtain the position information between mark and object more precisely. The distinguishing strategy is designed as follows:

According to Fig.14(a), we find that the B value of the box is higher than G and R values, and the G value is higher than the R value. So the limitation condition is B>G>R to distinguish similar objects after the recognition process.

According to Fig.14(b), we find that the G value of beer is higher than B and R

values, and the interval of BRG is within 30-40, so the limitation condition is G>B, G>R, G<40 and B>30 to distinguish similar objects after the recognition process.

According to figure 14(c), we find that the R value of the box is higher than the G value, and the G value is higher than the B value, so the limitation condition is R>G>B to distinguish similar objects after the recognition process.

According to figure 14(d), we find that the R value of the box is higher than the B and G values, and the interval of BRG is within 15-27 compared to the BGR value of the toothpaste box, so the limitation condition is R>G, R>B and R<27 to distinguish similar objects after the recognition process.

According to Fig.14(e), we find that the BRG value is very stable and each element has a relatively fixed interval. The R value of the box is higher than B value and the B value is higher than the G value, so the limitation condition is R>B>G to distinguish similar objects after the recognition process.

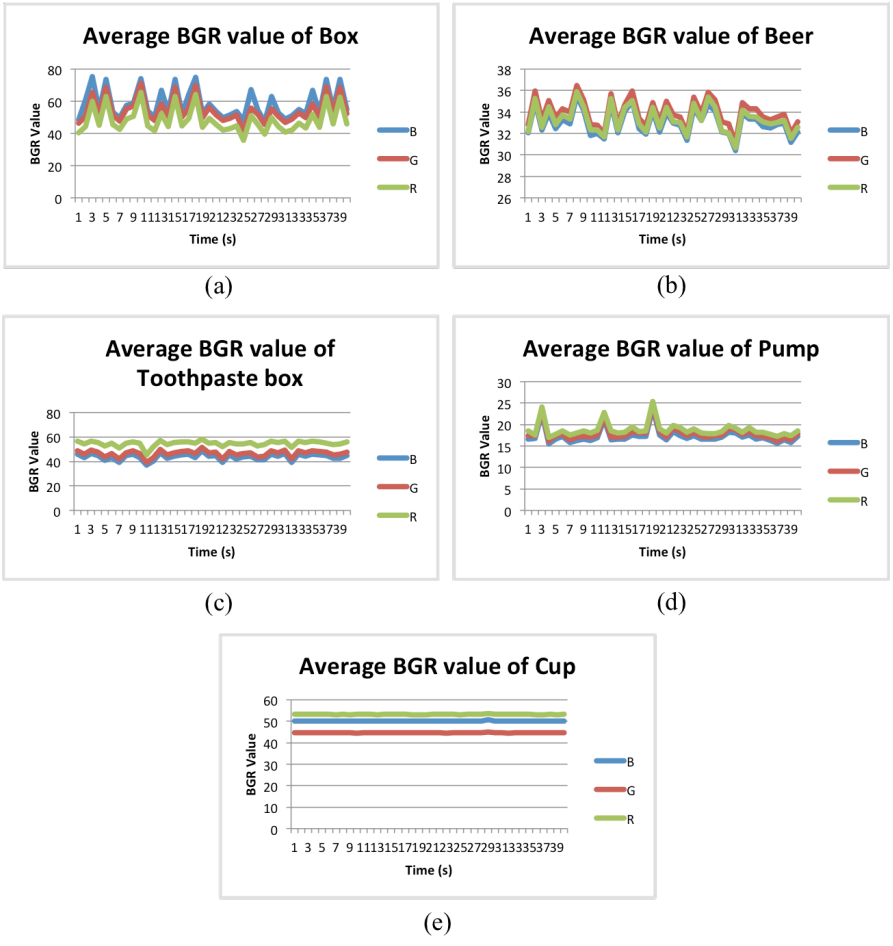
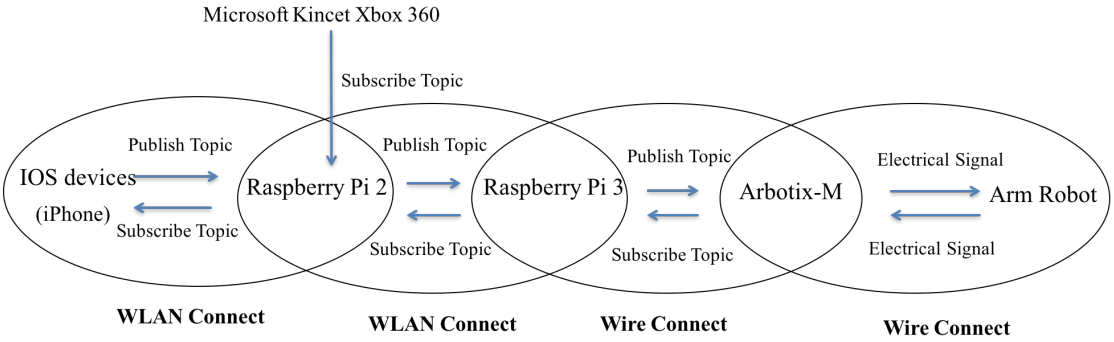


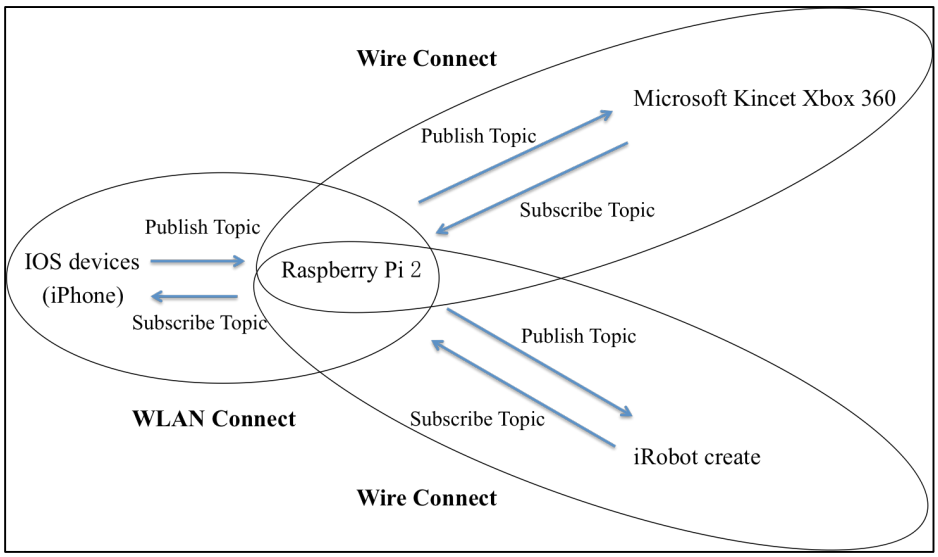
Figure 14: (a) Average BGR value of Box; (b) Average BGR value of Beer can; (c) Average BGR value of Toothpaste box; (d) Average BGR value of Pump; (e) Average BGR value of Cup

2.6 The Communication System

The communication among different devices of the system for controlling the arm robot via joint angular positions is shown in Fig.15. In this project, computer or ios device publishes the position information of object and arm robot to raspberry pi by wlan connect after image processing, and then raspberry pi publishes joint angular position to Arbotix-M after inverse kinematic calculation, and finally Arbotix-M transfers the joint angular position to electrical signal to run the robot arm. Vice versa, Arbotix-M microcontroller can read electrical signal and covert to real-time joint angular position, and then the raspberry pi subscribes topic from Arbotix-M and computer or ios devices subscribes topic from raspberry pi, and finally the program can check the status of the arm robot.



a)



b)

Figure 15: a) Communication system for manipulator, b) Communication system for mobile robot

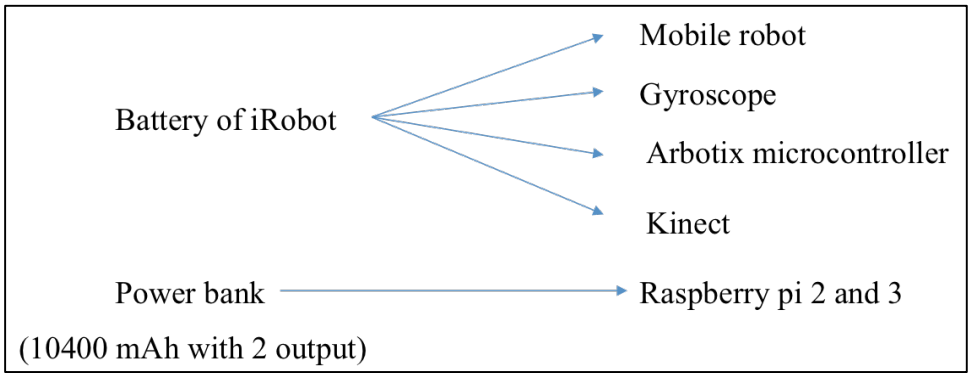


Figure 16: Power system for robot

In the robotic system, the battery system divides to two parts shown in the Fig.16: the battery of iRobot provides the power for the mobile robot, the gyroscope, the Arbotix-M and the Kinect; the power bank provides the power for two raspberry pi.

3 Experimental Evaluation of the System

In this section, we design some experiments to test the performance of the whole system based on the opinions of users for this robotic system. In the experiment, the users will choose one object in the interface, and then give the command to the robot, so the robot can pick up and deliver this object to the user. In the experiment, we set an obstacle and 3 interfering objects and 1 goal object in the table, so we can test the recognizing performance of the robotic system. We record the total time and the delay time in a trial. The experimental environment (setup) is shown in Fig.16.

Figure 16: The experimental environment

In the experiments, we asked 10 healthy human subjects (graduate engineering students) to test our system, and the results and the analysis are shown as follows:

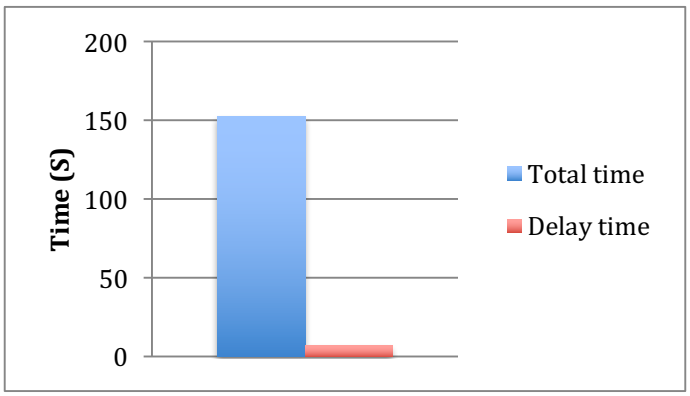


Figure 17: The mean (n=10) total and delay time of the robotic system for a trial

In Fig.17, the results show the total operation time and delay time of one picking and placing process. The delay time is the time after the user giving the command and the time when robot reads the RGB image from Kinect after it arriving the picking-up point. The average total time is around 150 seconds. The main problem for the large process and delay time is due to three reasons: first, the experiment environment includes some dynamic obstacle and the resolution of map is also a limitation during the navigation; second, the minimal distance for the Kinect sensor is 50cm, so the robot needs to move very slowly when it is closed to the user; third, because of the low resolution, the robot needs more time to adjust itself to a good picking-up point that the object is within the workspace of manipulator.

Table 1 is showing the task fulfillment during the experiment. The results show that the robotic system can recognize, pick, hand over and avoid obstacles correctly.

Table 1: Correctness in task accomplishment of the robotic system

	The robot can recognize the right object	The robot can pick the right object	The robot can handover the object to the human	The robot can avoid obstacles
Yes	9	9	9	10
No	1	1	1	0

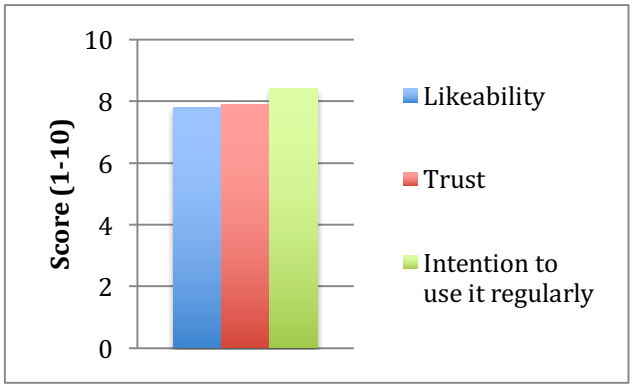


Figure 18: The mean (n=10) feedback scores of the users

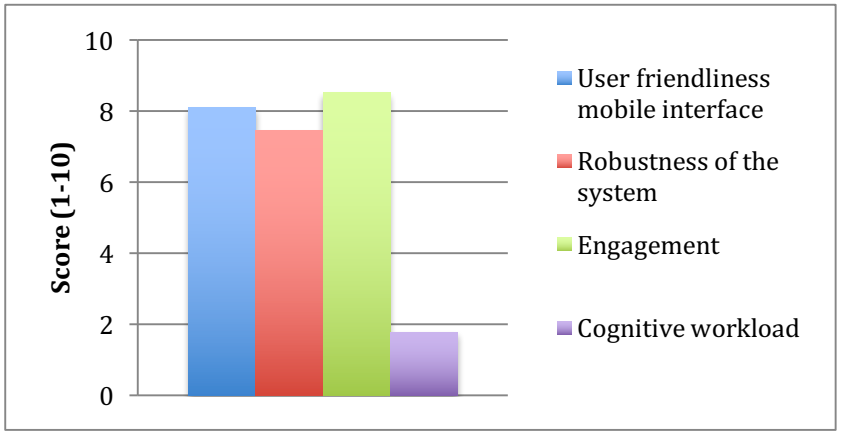


Figure 19: The mean (n=10) feedback scores of the users

The Fig.18 and 19 show the opinions of the users for this robotic system. One point needing to pay attention to is the score of the cognitive workload. In this part, score 1 means the best, and 10 means the worst. The score of this case is less than 2, and it means that the system results in low cognitive workload.

In this experiment, the system is now evaluated by healthy subjects, but it will be applied for disabled persons who cannot move easily and the system will be evaluated by actual disabled persons or patients in future.

4 The Conclusions and Future Works

Overall, the final prototype achieves the goal that the robot can deliver object remotely and automatically according to the requests of the human. The system can navigate according to known map and position information of multi objects, and recognize and deliver 5 objects one by one with different colors and shapes. It solves the problem statement and provides a convenient, interactive and smart robot system

to the user. Although it fulfills its desired role, it can be even better, and there are some goals of the next phase work as given below:

i. More accurate recognition algorithm to recognize objects with more complex color and shape.

ii. Using 3D SLAM technique to make map and recognize object simultaneously.

iii. The robot can complete object searching automatically.

iv. More function, such as, picking multi object at one time; the height of manipulator and Kinect can be adjustable for different tasks, etc.

5 Acknowledgement

I at first acknowledge Prof. Vikram Kapila of the Department of Mechanical and Aerospace Engineering, Tandon School of Engineering, New York University, Brooklyn, NY for kindly accepting me to work in his laboratory and providing me all the necessary resources for this project. Then, I acknowledge Dr. Mizanoor Rahman of the Department of Mechanical and Aerospace Engineering, Tandon School of Engineering, New York University, Brooklyn, NY for his instructions and guidelines for carrying out the project activities and preparing this report. I also acknowledge Mr. Jared Frank, Mr. Ashwin Raj Kumar and Mr. Saiprasanth Krishnamoorthy of the Department of Mechanical and Aerospace Engineering, Tandon School of Engineering, New York University, Brooklyn, NY for their technical supports. I also acknowledge the cooperation of all the subjects who participated in the experiment.

Reference

1. Grisetti, Giorgio, Cyrill Stachniss, and Wolfram Burgard. "Improved techniques for grid mapping with rao-blackwellized particle filters." IEEE transactions on Robotics 23.1 (2007): 34-46.

2. Grisettiyz, Giorgio, Cyrill Stachniss, and Wolfram Burgard. "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective

resampling." Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on. IEEE, 2005.

3. Fox, Dieter, et al. "Monte carlo localization: Efficient position estimation for mobile robots." AAAI/IAAI 1999.343-349 (1999): 2-2.

4. Thrun, Sebastian, Wolfram Burgard, and Dieter Fox. Probabilistic robotics. MIT press, 2005.

5. Craig, John J. “Introduction to robotics: mechanics and control.” Vol. 3. Upper Saddle River: Pearson Prentice Hall, 2005.

6. Viola, Paul, and Michael Jones. “Rapid object detection using a boosted cascade of simple features.” Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on. Vol. 1. IEEE, 2001.

7. Lienhart, Rainer, and Jochen Maydt. “An extended set of haar-like features for rapid object detection.” Image Processing. 2002. Proceedings. 2002 International Conference on. Vol. 1. IEEE, 2002.