King Fahd University of Petroleum and Minerals
Computer Engineering Department

Senior Design Project (COE485)
Semester 142

5/9/2015

Autonomous Person-Following Robot

Final Report

Team Members:

Adnan AlSinan	201015940
Abdulrahman Bataweel	200927570
Husam Habarah	200971590

Advisors:

- Dr. Muhamed Mudawar
- Dr. Yahya Osais

Table of Contents

Table of Figures

List of Tables

1 Introduction

Every day; we see shopping carts in the supermarket. Customers usually push the carts with both hands, and so, if the customer had only one hand or had to carry a child, then pushing shopping carts becomes a real burden.

Thus, in order to improve the situation and assist those with disability, we plan to develop a robot system that follows the user. Enabling disabled people to enjoy shopping with the need for human assistance (pick and drop on the robot).

1.1 How the project deals with the issue

People live in their lives in many modes. There is the young and adults mode where they are able to help themselves and help others. But there is certain stages in life that people needs help from others to carry their substances. It is embarrassing to ask help from anyone that might either happy accept, or might sadly refuse. This project aims to help those people to avoid asking others for help. A Robot will help in carrying there substances and follow them in a convenient way without disturbing their attention in shopping.

1.2 Project Impact on society locally and globally

1.2.1 Positive impact

Personal assistant robot is one of the promising areas where robotics could be put in practice. Nowadays, approaching "aging society" could make use of robots by helping people in their everyday life. To keep a better quality of life of elderly people, a robot could move with people like a partner and carry items without disturbing anyone by avoiding obstacles. Moreover, it is possible to use this robot to monitor and keep an eye at kids preventing them from hurting themselves or messing with electronic appliances. Beyond that, this robot could be used in different areas such as military, rescuing missions and medical applications.

2 Problem Statement

Autonomous robots have the capability of gaining information about the environment. They work without the need for human intervention for a long period of time. They can also adapt to changes in their surrounding environment.

In order to assist elderly or disabled people in supermarkets, our goal is to design and build a robot that is capable of following them, and carry weight for them, to substitute for pushing a shopping cart.

3 Background

3.1 Existing Solutions, Products & Research

3.1.1 Nippon Institute of Technology, Japan.

They were able to develop a robot shopping cart; the robot successfully recognized its user and followed him through the use of Laser Range Sensor (LRS). The sensor could detect objects in the distance of 4m and in the azimuth of 240 degrees.

However, this solution suffered a major flaw, when a third person cut across between the user and the robot shopping cart, the cart ended up following the third person. Minor flaws included, interference with the following procedure from obstacles such as shopping shelves and display tables.

Figure 1: Robot Shopping Cart

3.1.2 R&D Center, Toshiba Corporation, Japan.

ApriAttenda is a more sophisticated robot that uses cameras, LRF and ultrasonic sensors. ApriAttenda adopts a stereo vision system and additionally a LRF is mounted on it to enhance the performance of person following motion. The designed tracking system uses highly accurate measurement information by combining vision and LRF data according to the congestion level of movement space.

An experiment of person following was done using ApriAttenda using the sensor fusion method. And from this experiment, it was observed and confirmed that the robot can follow a person smoothly who moves quickly and randomly.

Figure 2: ApriAttenda Robot

4 Requirements and Specifications

4.1 Functional user requirements.

- The robot should be able to identify any person and be able to follow him/her.
- It should be able to recognize the person from different directions.
- It should be able to recognize obstacles such as aisles, counters, and people and avoid them.
- Extra feature: the robot should be able to start following after hearing

4.2 Non-functional user requirements.

- The robot should be able to carry at least 1 kg of weight.
- The robot can follow a subject within 2-5 M.
- The robot should not exceed a power consumption of 2100AMH.
- Response time should be within 2 seconds

4.3 Technical Specification.

- Identify subject to follow by camera.
- Following subject identified.
- Robot can carry up to 1 kg.
- Movement speed is 6 km/h to Follow-up human at maximum speed of 6 km/h.
- Keep distance of <2 meters to the followed person.
- Usage of distance sensors to avoid bumping into the user
- Height between camera and robot is 1.5 -2 meters.

5 System Design

5.1 Solution Concept

So the stated problem is, to create a robot that is able to follow its user and carry weight for them. That by itself, can be divided into two sub-problems, first is person detection, and second is person following. So, in all three solution concepts mentioned below, we are going to show how each one generally solves the problem.

5.1.1 General approach of solving the stated problem.

Description:

The approach proposed consists of using camera to detect the user of the robot, and distance sensors to measure the distance between the user and the robot.

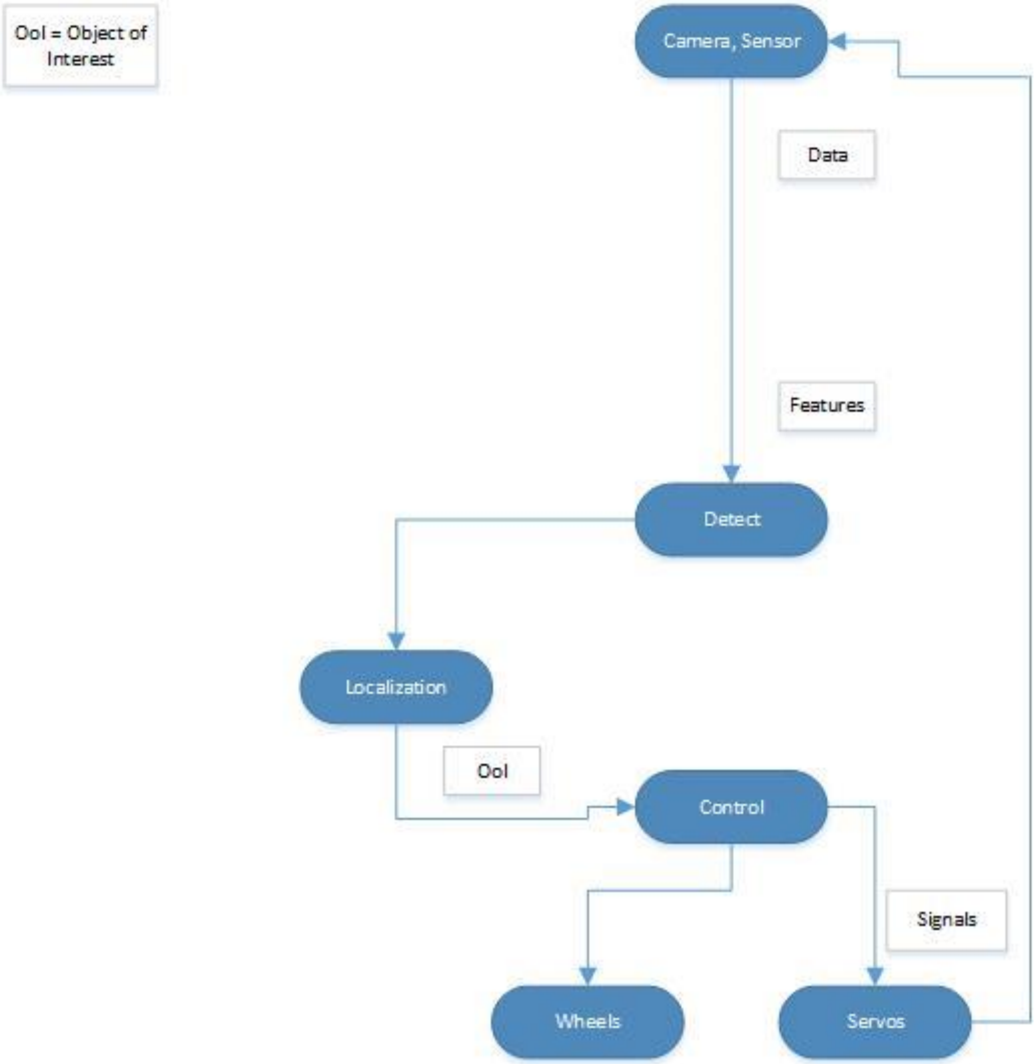


Figure 3: General Approach

5.1.2 Description of used/developed algorithms

- Image processing algorithm

In this project, a cascade classifier would be used that is a cascade of boosted classifiers working with *haar-like features* which are digital images features used in object recognition based on Haar wavelets rather than the usual image intensities. This classifier is trained with hundreds of the object of interest called positive images and negative images which do not contain the object.

After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier outputs a "1" if the region is likely to show the object (i.e., face/car), and "0" otherwise. To search for the object in the whole image one can move the search window across the image and check every location using the classifier. The classifier is designed so that it can be easily "resized" in order to be able to find the objects of interest at different sizes, which is more efficient than resizing the image itself. So, to find an object of an unknown size in the image the scan procedure should be done several times at different scales.

(More details about the image processing algorithms in appendix A)

5.1.3 Alternative approaches and algorithms.

Approach 2:

The alternative approach would use RFID technology in order to detect the person.

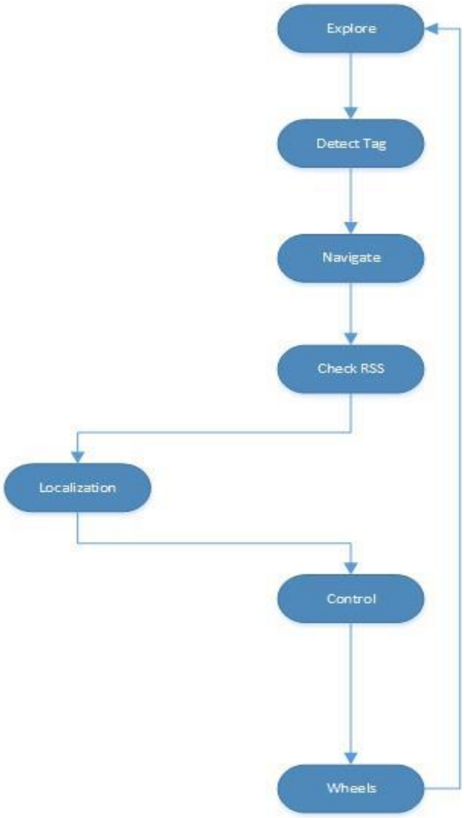


Figure 4: Alternative Approach # 2

Approach 3:

The third alternative approach is just using an infrared sensor (IR) to detect the person and follow him. However by just using a sensor, this approach is far beyond from being autonomous or smart. This approach relies directly on infrared sensor to detect the person and send commands to move the cart. However as mentioned previously, this approach fails when a third person cuts in between the robot and the user.

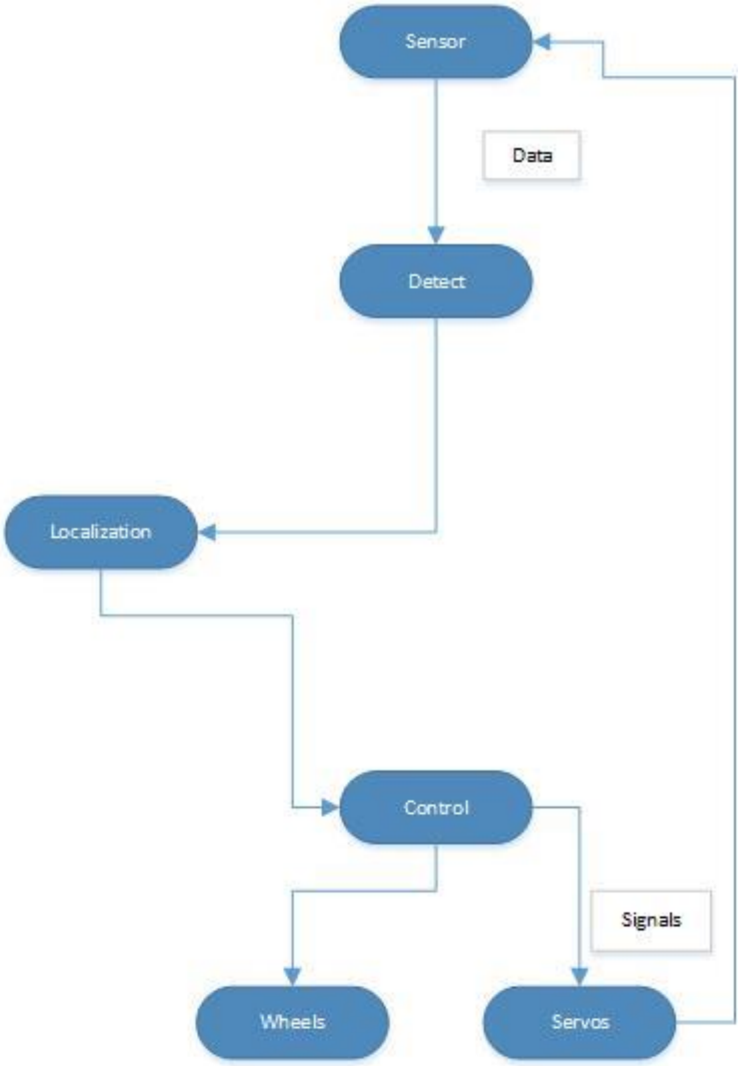


Figure 5: Alternative Approach # 3

5.1.4 Comparison, and selection criteria

Table 1: Approach Comparison Table

	Criteria					
Approach	Range	Accuracy	Invasiveness	Processing Time	Deployment	Price
RFID, with active tags	4 meters – 50 meters	Very accurate	Need to wear a tag or a device	Depends on the type of the tag	Two sides, on robot and on user	$ 500 - $ 2000
IR	1 meter – 3 meters	Not accurate, a third person might cut between.	No	Negligible	On the robot	$15 - $ 30

Camera and sensors	No specific range; dependent on the sensors	90%. [1]	No need to wear a tag nor to carry a device	the average processing per frame is about 51.76 ms.[2]	Everything will be on the robot.	Camera $300 - $400 Sensor $6 - $10

5.1.5 Sub-function identification

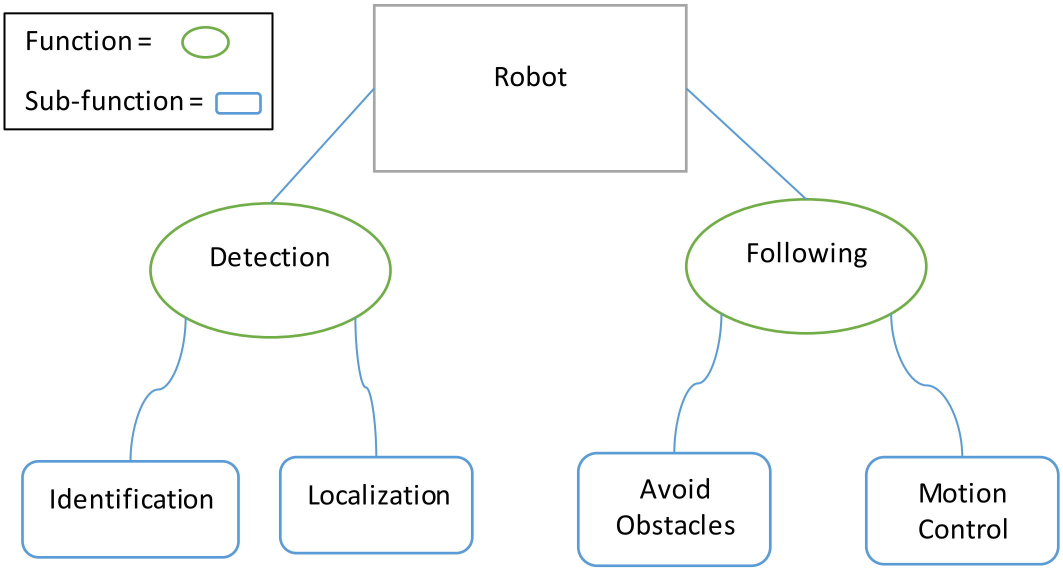


Figure 6 Sub-function identification

5.2 Architecture

5.2.1 System architecture and components

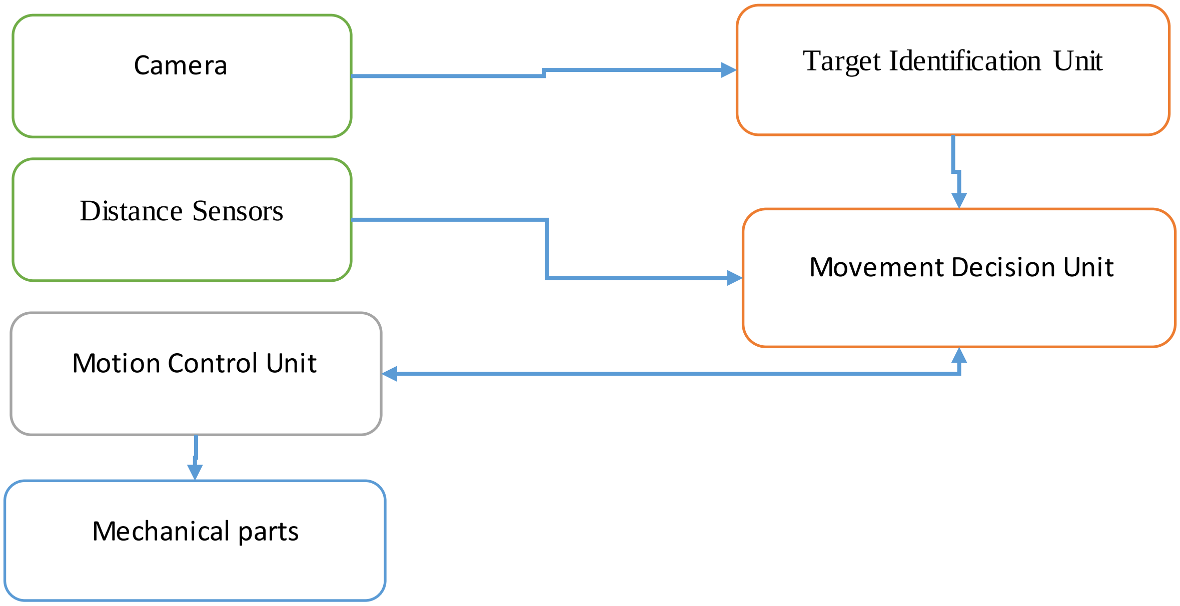


Figure 7: System Architecture

5.2.2 Alternative architectures

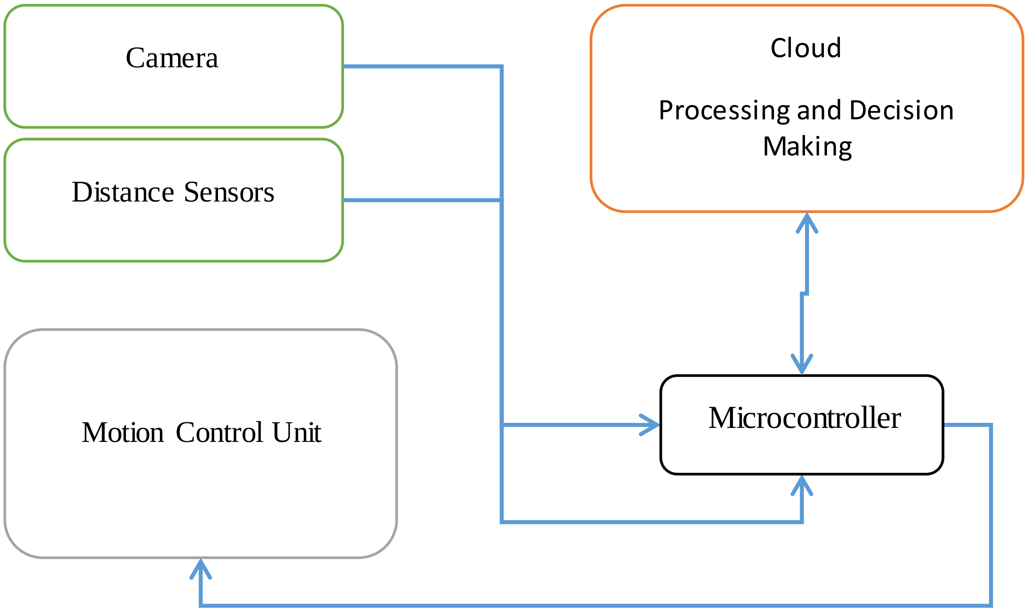


Figure 8: Alternative Architecture

5.2.3 Comparison, and selection criteria

Based on the following criteria, the first architecture was chosen despite the fact it require more processing, but the problem of power consumption can be dealt with by using a power bank. Because:

- Requiring internet communication is a major limitation.
- Not all the region is covered in terms of cellular network if Wi-Fi was not available.
- More cost will be needed for architecture 2.

Table 2: Architectures Comparison

ARCHITECTURE/CRITERIA	PROCESSING TIME	POWER CONSUMPTION [1]	RESPONSE TIME	ADDITIONAL COST
ARCHITECTURE 1 (WITH TARGET IDENTIFICATION UNIT)	Relative to hardware component	CPU-Intensive resulted in 608.708 mW	Negligible (connection is direct to component)	None
ARCHITECTURE 2 (WITH CLOUD)	Depends on the cloud	3G Download will result in 947.515mW	Dependent on the internet connection + signal coverage	Cellular network ; 1 GB/month 79 riyals

5.2.4 Hardware vs software components

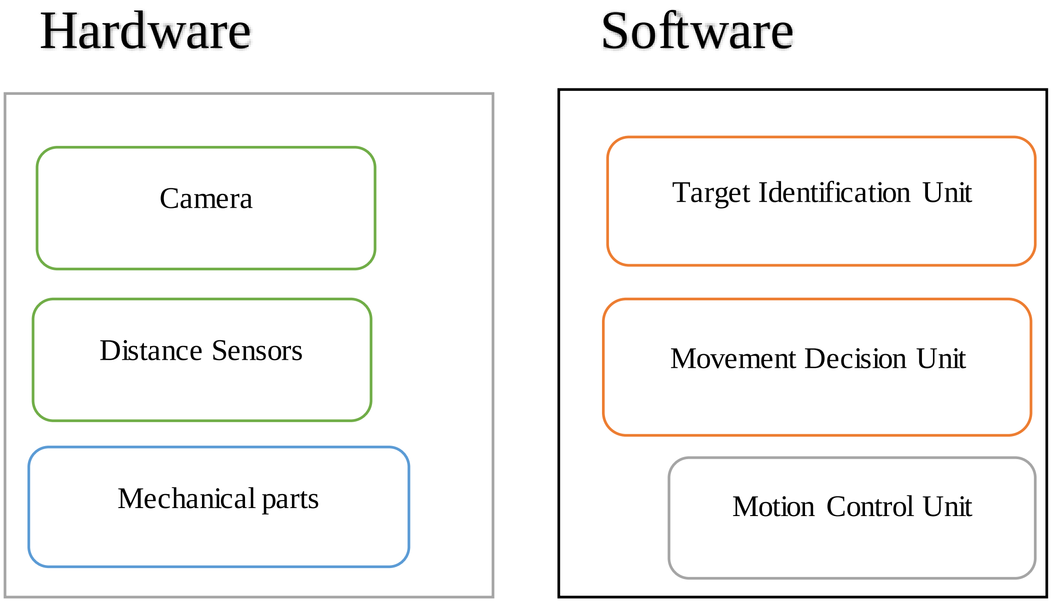


Figure 9: Hardware vs Software

5.2.5 Functions of each component

- **Camera**: Image input

---

[1] Profiling Power Consumption on Mobile Devices

- **Distance Sensors**: To measure the distance between the user and the robot, for the purpose of knowing whether he moved or not.
- **Motion Control Unit**:
  - Provide signals to move the robot based on the output from the movement decision unit
  - Control the servos
- **Mechanical Parts**:
  - Like wheels, servos.
- **Target Identification Unit**:
  - Receives the image from the camera and extract features of the object of interest in it.
- **Movement Decision Unit**
  - Decide the movement of the robot (whether it goes right, left, forward or backward) based on the sensors and image processing results.
  - Keeps the robot line of sight synchronized.

5.3 Component Design

5.3.1 Motion Control Unit

For the **motion control unit** we're going to use the T'Rex Robot/Motor Controller

The T'Rex controller from DAGU is an Arduino compatible robot controller designed to power and control servos and brushed motors. This controller combines an Arduino development board with a dual FET H-bridge motor driver. The heart of the board is an ATMega328P AVR microcontroller

The T'REX controller is rated for a **maximum voltage of 30V** and can handle currents in excess of **40A per motor**. A set of screw terminals are provided for **the battery connection** as well as for **connecting motor outputs.** The Dual H bridges are rated for stall currents of 40A per motor and average currents of 18A per motor.

Features:

- 6V -30V operation with built in solid state power switch
- Programmable with the Arduino IDE (ATMega328P, 5V @ 16MHz)
- Dual FET "H" bridge rated 18A with self-resetting PTC fuses
- Electronic braking and current monitoring for each motor
- 3-axis accelerometer provides angle and impact detection
- Auto-detects RC, Bluetooth, or I^2C control
- Voltage translation on I^2C interface
- 6x Servo Outputs
-

5.3.2 Distance Sensors

These sensors will be used to detect the distance between the person and the robot itself in order to detect movement. Another approach is to use image processing to detect the movement which is not reliable and difficult to maintain. Furthermore, the distance sensors would be used to avoid obstacles.

In this project it makes more sense to use distance sensors rather than relaying completely on the image processing. There are two available options to use as distance sensors; the first one is to use ultrasonic sensor or laser-range-finder. In terms of cost, ultrasonic sensor is cheaper than laser-range-finder and relatively using ultrasonic sensor is easier. Subsequently, ultrasonic sensor would be used for both detecting the distance between the robot and the subject and for avoiding obstacle.

Ping ultrasonic sensor which has only has three connections, which include Vdd, Vss, and 1 I/O pin. The 3-pin header makes it easy to connect using a servo extension cable, no soldering required.

5.3.3 Target Identification Unit

For the Image processing and movement decision units, we had four options; an android device, an iPad/iPhone device, Beaglebone black, Raspberry pi 2.

Based on the comparison in table 3 and 4, in terms of cost, at first glance, raspberry pi or Beaglebone look like they are the cheapest however considering the camera which is SR 180 making the price SR 330 and 370 respectively. Whereas an android phone with a price of 500 to 1450 or even more could give a room for improvement as there is many phones with different processing power. Also, the utilities in the phone could become handy such as the built-in camera, wireless, cellular network and the built-in sensors. Unlike iPhone, android libraries concerned with this project are more documented and easier to use, and to develop an iPhone app, a Mac environment must be used because of Xcode that is the IDE used to develop iPhone apps which can only runs in Mac machines, and such environment does not exist in KFUPM. Based in all that, it is reasonable to use an android phone as a prototype.

Table 3: Image Processing and Movement Decision Units options comparison

Option	OS	CPU	Storage	RAM	Camera	BT	Connectivity	Cost	Development platform
Beaglebone black	Linux	Dual-core 1.3GHz	4GB eMMC, and micro SD	512MB DDR3	Not built in	No need	Ethernet	SR 190	C, C++, Python, Perl, Ruby, Java, or even a shell script
Raspberry PI 2 B	Can run Linux/ windows	Quad-core 900MHz	Micro SD	1GB	Camera interface	No need	Ethernet	SR 150	C, C++, Python, Perl, Ruby, Java.
Android device	Android lollipop /KitKat	Quad-core 2.3GHz Dual-core 2.3GHz	16 GB	2GB	8 MP	v4.1, A2DP, apt-X	Wifi, GSM	SR 500-1450	Free to develop. Uses Java as a programming language for development
IPhone 5s	iOS 8.1.3	Dual-core 1.3GHZ	16 GB	1GB	8 MP	v4.0, A2DP	Wifi, gsm	SR 2225	Restrict development, must pay for developer ID. Languages are restricted to Swift and objective C.

Table 4: Cameras options comparison

	Criteria	
Option	Frame/second	Cost
Android built-in Camera	1080 @30 fps	Included with the android device
IPhone built-in Camera	1080 @30 fps	Included with the iPhone device
Logitech c920	1080 @ 5(at jpeg format)-30(at h264 format) fps	SR 400
Logitech c310	720 @ 15 fps	SR 300

Two approaches were suggested to tackle the problem of detecting and tracking the target. The first one was to use the color of target's cloths to identify him or her. The other method is to use algorithms that use features to identify the target.

For the first choice which is using colors to detect the target, it began with developing an android app that could detect objects of a specific color. The way it works as in Figure 10 is to touch the region of interest, after that, the location of the object in the photo as well as the number of objects of the same color will be showed. So here a decision was made to identify the target as the largest object and to ignore the other objects. To test this system, we got it into the field where it could possibly be used and the results were as follows, it was so fast to run in the mobile phone and it did not consume a lot of power. However, it was not reliable since any change in the angle of the camera could let the app detect different color due to the brightness and any dark color would be considered black which made the app think any switched off monitors in the lab as objects. Furthermore, it is not practical to ask the user wear a specific color add to that some t-shirts contain different colors.

The other option was to use an algorithm that could identify humans rather than colors, there are different algorithms used in that matter. The way they work, first they need to have a training data that is consist of positive images and negative images, in detecting humans, positive images could be images of human in different angles and illumination and the negative images should contain images that do not have humans, just a background it could be an empty class or a street etc. . Next features will be extracted from the images and to be used in a classifier which can based in the training to detect humans. This was tested in different degrees of illuminations and different angles and the results were robust and reliable enough as Figure 14 illustrate.

After detecting the target, the location of the bounding box is extracted as coordinates (X, Y); the value of Y is discarded because it is desired to locate the person if he or she goes right and left. And the value X is given as input to Movement Decision Unit.

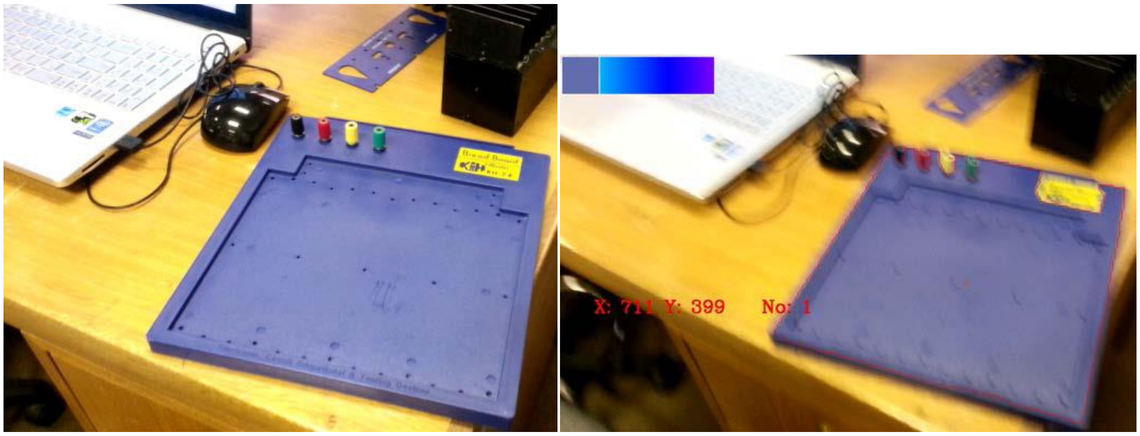


Figure 10 Color app detection; first, the color is touched, and then details will be showed

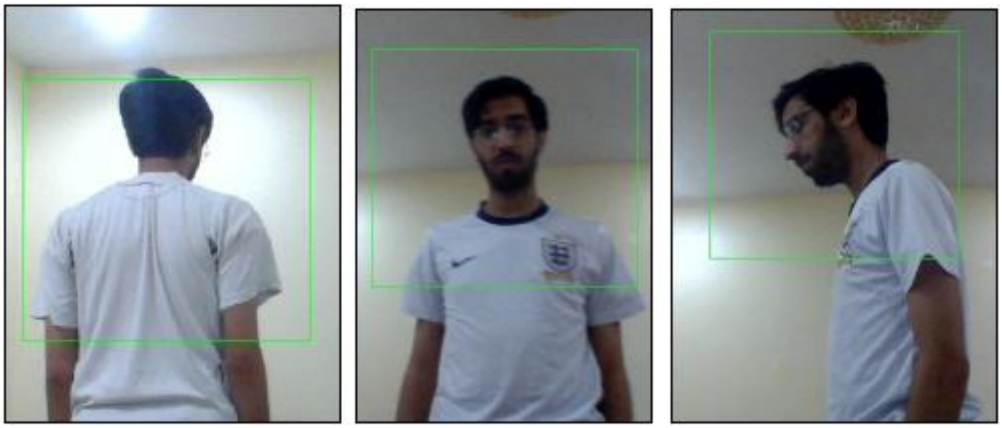


Figure 11 Detecting human based on features, from different angles

5.3.4 Movement Decision Unit

This is the main component of this project, it just like the brain and the other components are senses and muscles. This component is customized to this specific project. This unit takes two inputs:

- Result of Target Identification Unit, donated as X (location of the person in the frame).
- Distance reading values, donated as d (distance between the target and the robot in CM).

The value of X is generated every captured frame whereas the value of *d* is generated every 100 MS.

The frame is divided into three regions center, right and left as figure 12 shows. The way it works is to check the value of X, if the value of X is in the left region, a command is sent to motion control unit to going right until the target is in the center region. And the same method is used if the target in the right region.

The aim of this approach is to get the distance reading of the target as the sensor should be somewhere nears the camera.

As mentioned above, every 100 MS there is a new reading sensor *d*, if *d* is more than a specified value e.g. 140 CM a command is sent to motion control unit to go forward until it is equal or below that specified value.

Figure 12 The regions of the frame

5.4 System Integration

5.4.1 Interfaces

For system Integration, **Movement Decision Unit** should receive two inputs in order to make follow decision. First input is an integer value represents the x-coordinate of the corner of object of interest. This is received from **Target Identification unit.** Second input is integer value of the distance between the robot and the object of interest which in this case should be in the middle of the robot. It is received from **distance sensors**.

By compiling those two inputs in algorithm it should provide a decision whether the robot will move or not, and in which direction to move, how much distance should be covered to keep the specified distance between robot and object.

Communication between Movement Decision and Target Identification is done inside the software, while communication between ultrasonic sensors and Movement Decision through Bluetooth channel using socket.

Motion Control Unit receives one of four characters to control robot's movement: "f' forward, "d" right, "a" left, and "q" to stop. There is one more character for reverse, "r" but not used in this project.

The distance value is used to determine if there is an **obstacle** in the path specified or not, this will be processed by the Decision Making Unit by using provided inputs from **distance sensor**.

Distance sensors are connected to T'Rex using wires for power, ground and signaling to provide data to the Follow-Person algorithm.

5.4.2 Sequence Diagram

When powering up the robot, it is an idle state. The tracking mode should be enabled somehow it could by pushing a button or voice command etc. When the tracking mode is enabled, the camera start capturing frames and they are given as input to target identification unit aka T.I.U. Then they should be processed to extract the location of the target in the frame, the result should be directed to movement decision unit to decide how the robot should be moved. According to the result of the logic inside movement decision unit, commands are sent to motion control unit. After that, motion control unit generate signals to the motors, as illustrated in figure 13.

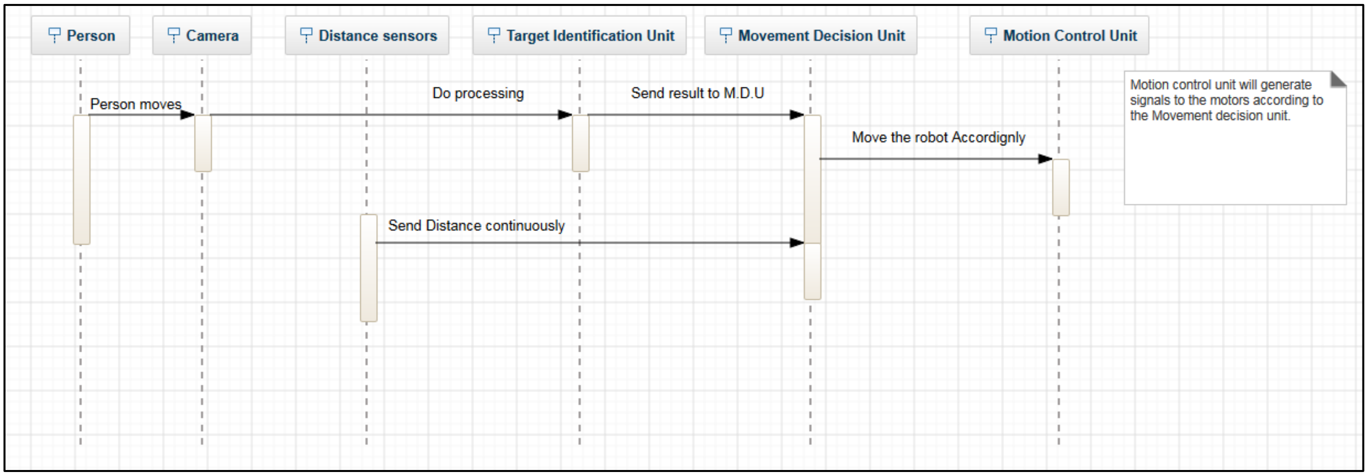


Figure 13: Component sequence diagram

5.5 Design Evolution

5.5.1 Physical Design

For the Robot physical parts, it was planned to use a stick holder to carry the camera in horizon with human height. However, that was not stable and the weight of the device and camera caused unstable situation of the stick. The selfie stick was useful to hold the mobile; Figure 14 shows how the implementation was.

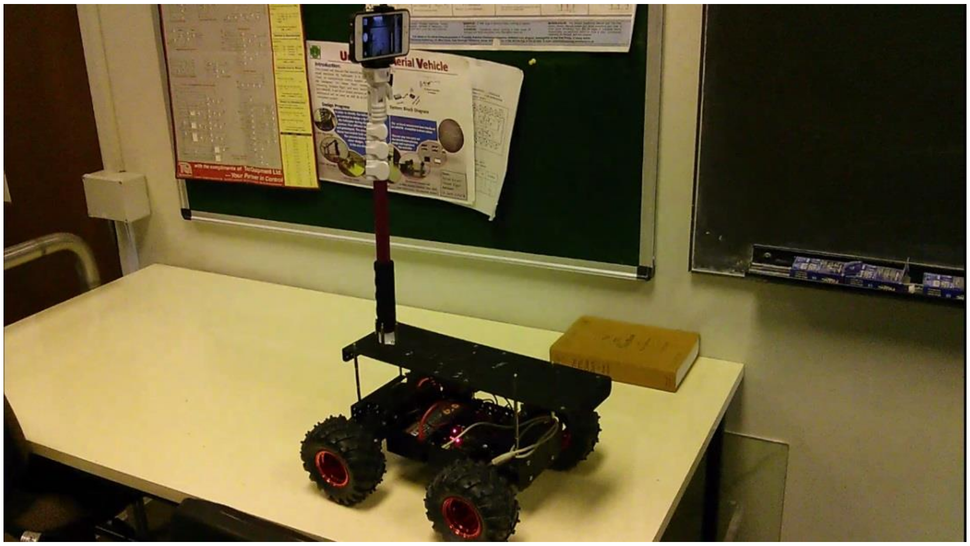


Figure 14 Initial Mobile Holder

The problem was the fast response of motors to the current. There was no gradually speed slowing or start. The robot jumps when forward command is issued which make the whole stick and device bounce. In addition, it was not reliable to have long stick and on top of it a mobile while we can still get the function needed working while the camera on the surface of the robot.

Design then become much stable when using another part of mobile holder. A car's mobile holder is used therefore to mount the mobile as illustrated in Figure 15.

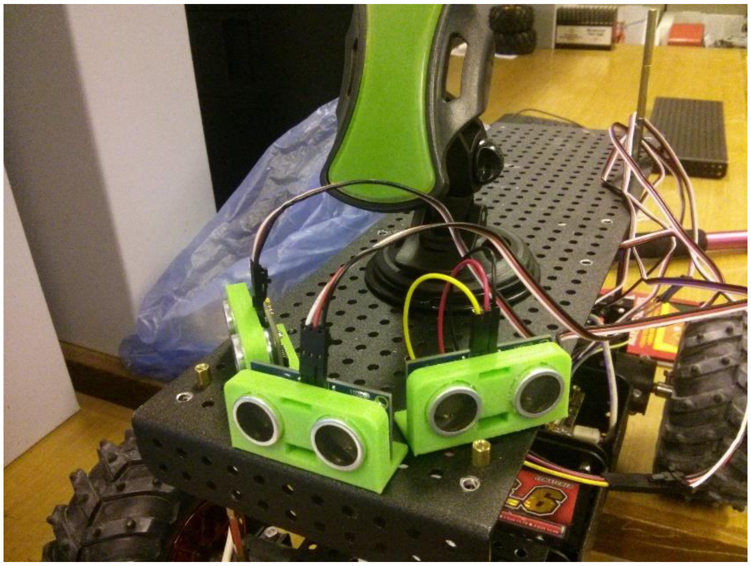


Figure 15 Car Mobile's Holder and Ultrasonic Sensor are placed

On the other hand, a three Ultrasonic Sensors are used to provide distance for robot functions. This is important in avoiding obstacles in three angels and gives the right distance to the person being followed.

To hold the Ultrasonic sensor a 3D design of Ping Ultrasonic is fetched from the Internet and printed in Dhahran Fab Lab, it took three hours to print four ultrasonic holders.

A screw is used to place the holders to the robot's body.

In the final design, a vertical metal holder was added to hold the car's mobile holder on top; to provide more height for the holder in horizon with human height as illustrated in Figure 16.

Figure 16: Final Robot Design

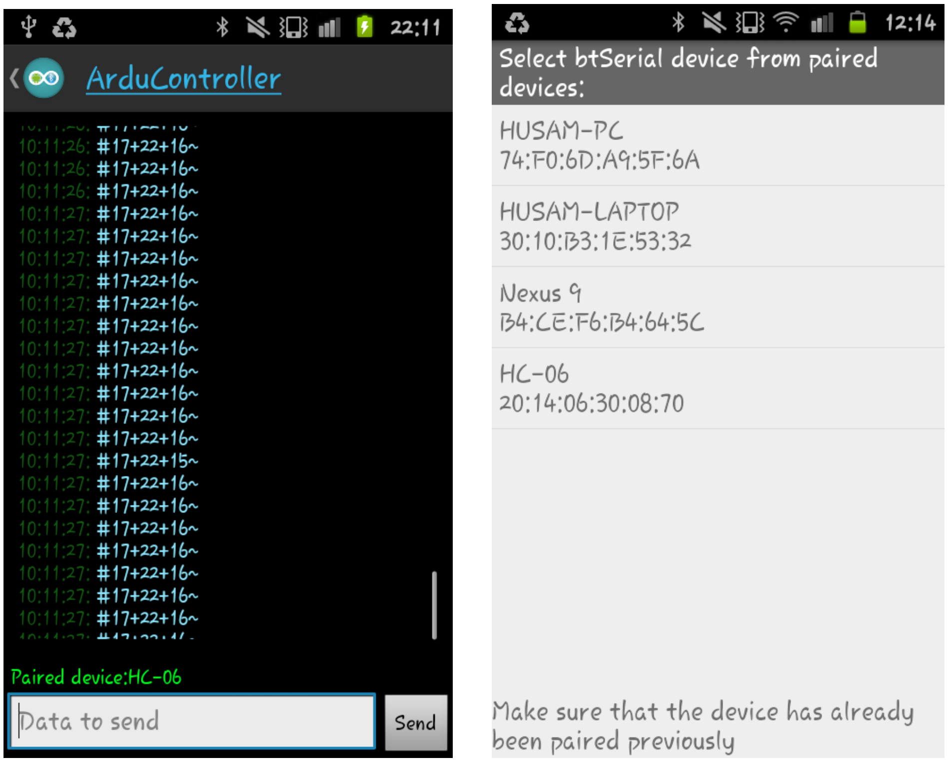
5.5.2 Software Design

- Motion Control Unit:

One command for the robot to control its movement and speed was implemented from the code received while other functions were not tested or implemented for use. Robot was capable of moving in one direction which is forward. Other movement choices are added according to the project need. Left, right, reverse and stop are implemented by controlling the motor rolling direction. For left turning, left side of the robot "which contain two motors" will troll in forward direction while right side of motors will roll backward. Right is therefore the opposite of left implementation where the right side of the robot will forward roll and left side of motors will roll backward. Stop is implemented by setting the motor's speed for all motors to zero.

- Android app:

Initially, the software used in the robot itself was for testing meaning that it must be connected to a PC and by serial a commands were sent to test the functionally of the robot such as going forward, backward and changing direction as well as controlling the speed. After that, an android app was used to that testing, it works by sending a characters as figure 17 shows. Next, a more user-friendly was used that uses buttons to send commands and retrieve sensors values, illustrated in figure 18. Afterwards, image processing was added which made the software more interesting and efficient. At the end, the software was able to communicate with the motion control via Bluetooth and do image processing while maintain a simple interface as indicated in figure 19.



```
10:11:26: #17+22+16~
10:11:26: #17+22+16~
10:11:26: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+15~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
10:11:27: #17+22+16~
```

Paired device:HC-06

Select btSerial device from paired devices:

HUSAM-PC
74:F0:6D:A9:5F:6A

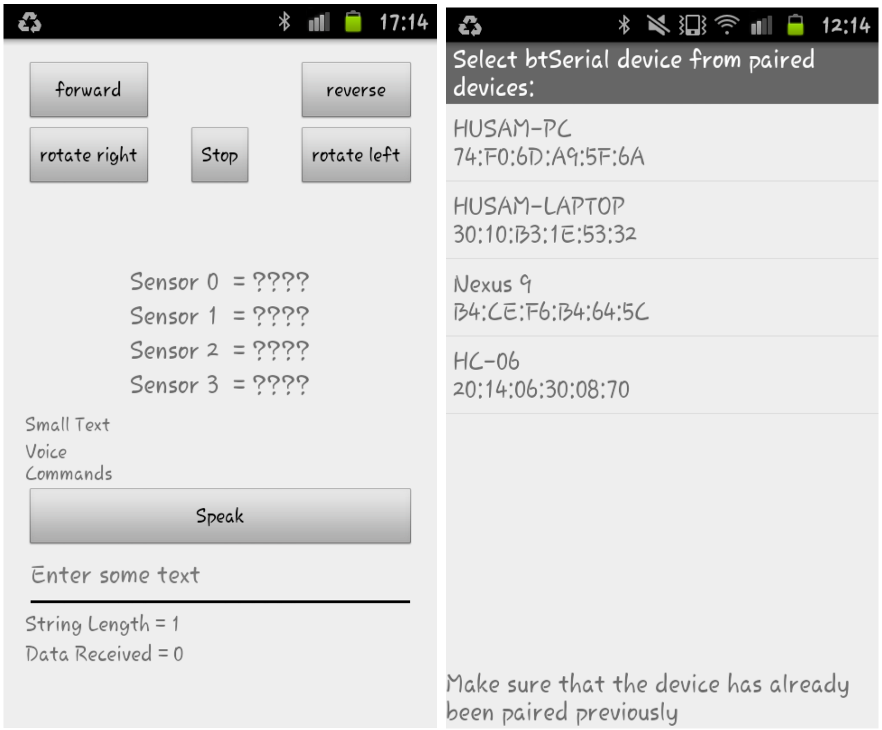
HUSAM-LAPTOP
30:10:B3:1E:53:32

Nexus 9
B4:CE:F6:B4:64:5C

HC-06
20:14:06:30:08:70

Make sure that the device has already been paired previously

Figure 17: inital BT software

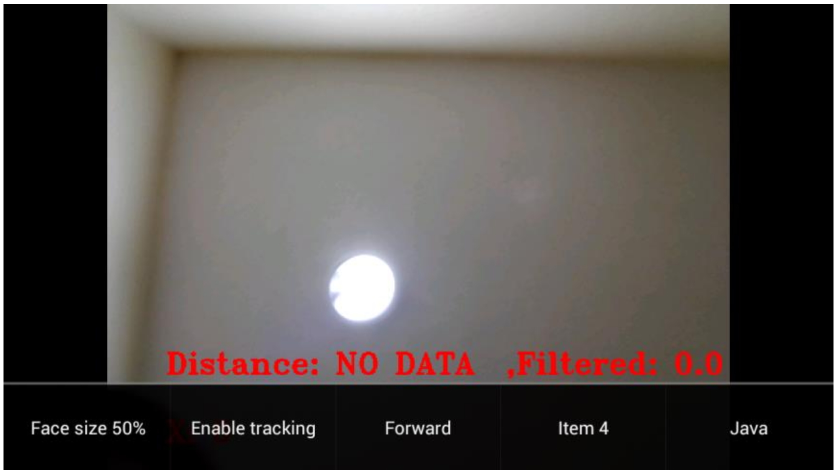


Sensor 0 = ????
Sensor 1 = ????
Sensor 2 = ????
Sensor 3 = ????
Small Text
Voice Commands
Enter some text
String Length = 1
Data Received = 0

Select btSerial device from paired devices:
HUSAM-PC
74:F0:6D:A9:5F:6A
HUSAM-LAPTOP
30:10:B3:1E:53:32
Nexus 9
B4:CE:F6:B4:64:5C
HC-06
20:14:06:30:08:70

Make sure that the device has already been paired previously

Figure 18: Better app for simpler controlling



Distance: NO DATA ,Filtered: 0.0

Figure 19: Final version of the app used for tracking

5.5.3 Functionality

As described in the previous section, the robot has started with minimal functionality to the extent it needs to be connected to PC to move it. Next, after developing the app and adding the Bluetooth module, it is possible to control it via Bluetooth and use the phone as a remote control. Next, when the image processing was added to the app, the robot can change direction based on the person's location in the frame but yet cannot go forward. Afterwards, the sensors were added and the readings became available, the robot can go forward based in the readings of the sensors.

6 Testing, Analysis and Evaluation

6.1 Testing methodology and results

The project was tested in three ways:

- Component/Unit testing: testing each component alone.
- Integration testing: any integration between components gets tested.
- System testing: test the complete system to make sure it's working.

Component/Unit testing:

- Android Bluetooth App: Involved testing the android application with the DAGU Bluetooth module on the T'Rex controller to ensure connectivity.
- Color Detection App: tested the app, to check if it can detect objects with certain color.
- Face (feature) Detection app: tested the app individually to check if it can detect faces.
- Motion Control (basic movement functions): tested forward, backward, turn left and turn right functions using serial interface.
- Voice Recognition (Speech to Text and Text to Speech): tested the android app to ensure that it can reply after hearing certain commands.

Integration testing:

- Bluetooth with Motion Control: verified and tested that the microcontroller can receive commands via Bluetooth and execute them.
- Ultrasonic sensors values and Bluetooth: verified and tested that the Bluetooth app can receive the sensors values correctly.
- Face Detection, Bluetooth and Motion Control: verified and tested that the robot can follow the person's movement (turning right and left)

6.2 System *analysis* and *evaluation*

All system analysis and Evaluation has been done outdoors.

Respond time to human movement was acceptable to Real-Time system. Ultrasonic sensors provide continuous data to the system with time of less than 50 milliseconds.

These important functions are crucial for testing system responsiveness to the user movement and keep following function on track. Testing is mainly done by placing the robot behind a person and examines different type of human walk. Walk direction is changed to test the system response to the object movement as in real life and to test detection function. Distance between robot and object of interest is controlled by the sensor accuracy which is subject to some noise, this could be overcome by talking average to multiple reading of the sensor.

7 Issues

7.1 Problems faced

- Mounting the smartphone on top of the robot:
  - **Description:** we had two options to mount the smartphone on top of the robot. 1) Was to use a selfie-stick and then attach the phone to it. 2) To cut a servo-sized hole on the robot top cover, insert the servo down, 3d design and print a small servo gear that can fit in both the servo and the holder, then attach the phone.
  - **Attempts:** we have tried both approaches, the first one required us to design a base for the selfie-stick that can put on top of the cover, and because we lack in experience in 3D design, we did not continue with this approach. The same goes for option 2, which is cutting down a hole in the cover, making a suitable 3d printed servo gear that can fit.
  - **Solution:** At the end, we decided to go with a car phone holder, and put it directly on top of the cover.
- Synchronization:
  - **Description:** As discussed before, the distance sensors give readings every 100 MS whereas generating frames and processing them takes a lot longer than 100 MS causing latency and malfunctioning. This problem was popped up when doing turning and moving forward at the same time.
  - **Attempts:** an attempt was to decentralize the movement decision unit by having it merged into the motion control unit. And to make the sensors and capturing frame independent of each other, meaning as soon as there is a reading move the robot accordingly and the image processing will follow up. However, it did not work if the target had changed the direction very quickly.
  - **Solution:** The solution was to make decisions only when a frame is processed, so all the readings of the sensors before that are discarded. So, when a frame is processed and the target was found, only in that case make decisions whether to go forward or change direction or both.
- Power Failure:
  - **Description:** we faced a problem where the first T'Rex microcontroller suddenly stopped working after showing signs of smoke.
  - **Attempts:** we investigated possible sources of error; connections from the power source to the controller, the controller itself, and the USB cable to the controller. In the end, we came to the conclusion, since it can be powered using the USB cable, and it is not an issue from the power source connections, then it must be a problem with the microcontroller itself.
  - **Solution:** the old T'Rex microcontroller got replaced with another one that is functional.

7.2 Limitations and Constraints

- One constraint the robot has is that it cannot move freely in the carpet because the wheels were not designed to; and at the beginning, the team was testing the robot in the carpet that caused one of the motors to stop working. To avoid that, the subsequent testing was done outside or in the flat surfaces.

- Another constraint is, a relatively good lightening and illumination is required, because most of the image processing algorithm will not produce good and reliable results in such environment.

- A limitation that is the robot will follow any person in front of it, so if there are two persons in the sight of the robot, it will be alternating between them causing it to produce unpredictable behavior. A small modification to target identification unit can fix that by adding training phase before tracking the person. So after training it should follow that specific person. But this modification is put into future work.

- Another limitation is the left and right movement of the robot. It is different from regular vehicles. Forcing the robot to turn left or right require all the motors to move (one side opposite to another).

8 Engineering Tools and Standards

The tools and standards were chosen and used based on the following criteria:

- **Easy-to-use**: How hard to learn and how much time needed to master the tool.
- **Compatibility**: Is the specific tool compatible with different platforms.
- **Cost**: How much cost to use or to buy the tool.
- **Efficiency**: How productive the tools is, does contain a lot of errors or limitation.

The tools used in this project were:

- Microcontroller for controlling the motors:

  ❖ The **T'Rex Microcontroller:** is an Arduino compatible robot controller designed to power and control servos and brushed motors. This controller combines an Arduino development board with a dual FET H-bridge motor driver. The heart of the board is an ATMega328P AVR microcontroller. It was chosen because it is compatible with the available robot chassis and it is reliable and efficient, another option to it to use stock Arduino and motor driver separately which needs more connections and more physical space. And it uses Arduino software for programming making it easy to use.

- The physical design:

  ❖ The **Wild Thumper 6WD Chassis:** the robot chassis was provided to us by Dr. Mudawar along with the T'Rex Microcontroller; Ready-made component saved a lot of time because coming up with design and to implement it is lengthy task. And the case has a lot room of space for adding utilities.

- Android development:
  - **Android Studio:** this the new official IDE for Android application development provided by Google. It has new features such as the gradle. And it was used because it is more reliable than eclipse.
  - **Eclipse with ADT:** this is the old fashion of writing android apps, despite the fact, it might be easier to use than android studio but it is full of bugs and keeps crashing with no obvious reasons.

- Prototyping image processing algorithm:
  - **Visual Studio + OpenCV:** OpenCV is an open source library that contains many computer vision algorithms. OpenCV algorithms were written in C++ and they were tested on PC before using them in android phone. OpenCV is free to use unlike MATLAB. Because of these reasons it worked like that, first test the programs in C++ on PC and if the results were OK, implement them in Android.

  - **MATLAB:** is a high-level technical computing language and interactive environment for algorithm development. It might be easier to use for prototyping but to implement the corresponding algorithm in OpenCV or java is time-consuming. Furthermore, it is costs a lot to use it.

Used Standards:

- **Bluetooth**: a wireless technology standard for exchanging data over short distances. Another option was to use the USB port of the mobile phone and to connect it to as a serial to the Arduino controller. But it is not easy to use the USB interface in Android, and it requires a chip that converts USB to UART. The Bluetooth was chosen because it is easier to use and to implement. Another reason is that, because of the power-consumption of the software, it may be desired to connect the phone to a power bank.

9 Team Work

Working in this project indeed helped team members' ability to work in a collaborative environment and to finish tasks faster. Table 5 shows the tasks for the project

Table 5 Work Distribution

Member	Responsibilities	Contribution	Expertise
Abdulrahman	○ T'Rex Programming ○ Obstacle Avoidance ○ Motion Control Functions	○ Testing & Debugging ○ Android Speech Recognition	○ T'Rex (Arduino) Programming
Adnan	○ Image Processing & Detection ○ Color Detection Method ○ Feature Recognition Method	○ Testing & Debugging ○ Android Speech Recognition	○ Computer Vision
Husam	○ Communication between Android and T'Rex ○ Controlling the robot via Bluetooth ○ Android Speech Recognition	○ Testing & Debugging ○ Android Speech Recognition	○ Bluetooth communication

10 Conclusion

For this project, autonomous robot is capable to follow a person, although the robot can be used in many application, this product aims to solve a particular problem which is helping people carry their goods inside a market, making their lives easier.

This report identifies requirement, specification, advantages and disadvantages of the product.

Furthermore, team's members have learned how to design and integrate different components together and synchronize between data exchange. Developing android application using Android Studio.

11 References

1. Ardito,L., Procaccianti, G., Torchiano, M., and Migliore, G. (2013) Profiling Power Consumption on Mobile Devices , *The Third International Conference on Smart Grids, Green Communications and IT Energy-aware Technologies.*
2. Takafumi, K., Onozato, T., Tamura, H., Katayama, S., & Kambayashi, Y. (n.d.). Design of a Control System for Robot Shopping Carts. Retrieved February 1, 2015, from http://leo.nit.ac.jp/~tamura/pdf/1780.pdf
3. Sonoura, T., Yoshimi, T., Nishiyama, M., Nakamoto, H., Tokura, S., & Matsuhira, N. (n.d.). Person Following Robot with Vision-based and Sensor Fusion Tracking Algorithm. Retrieved February 4, 2015, from http://cdn.intechopen.com/pdfs-wm/5205.pdf
4. Tarokh, M., & Merloti, P. (2010). Vision-Based Robotic Person Following under Light Variations and Difficult Walking Maneuvers. *Journal of Field Robotics, 27*(4). Retrieved February 2, 2015, from http://onlinelibrary.wiley.com/
5. Cameron, S. (n.d.). *A comparison of keypoint descriptors in the context of pedestrian detection: Freak vs. surf vs. brisk*. Informally published manuscript, Standford University CS Department, . Retrieved from http://cs229.stanford.edu/proj2012/Schaeffer-ComparisonOfKeypointDescriptorsInTheContextOfPedestrianDetection.pdf
6. Human Walk Speed:
7. Tian, Q., Zhou, B., Wei, Y., & Fei, W. (2013). Human Detection using HOG Features of Head and Shoulder Based on Depth Map. *Journal of Software, 8*(9). Retrieved February 10, 2015, from http://ojs.academypublisher.com/index.php/jsw/article/view/9783

12 Appendix A

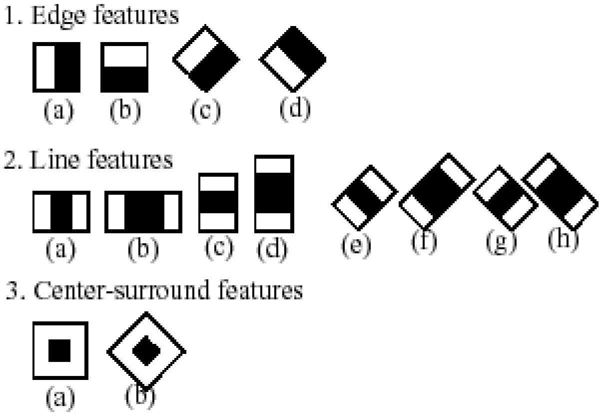
- **Image processing algorithms**

The object detector described below has been initially proposed by Paul Viola and improved by Rainer Lienhart.

First, a classifier (namely a *cascade of boosted classifiers working with haar-like features*) is trained with a few hundred sample views of a particular object (i.e., a face or a car), called positive examples, that are scaled to the same size (say, 20x20), and negative examples - arbitrary images of the same size.

After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier outputs a "1" if the region is likely to show the object (i.e., face/car), and "0" otherwise. To search for the object in the whole image one can move the search window across the image and check every location using the classifier. The classifier is designed so that it can be easily "resized" in order to be able to find the objects of interest at different sizes, which is more efficient than resizing the image itself. So, to find an object of an unknown size in the image the scan procedure should be done several times at different scales.

The word "cascade" in the classifier name means that the resultant classifier consists of several simpler classifiers (*stages*) that are applied subsequently to a region of interest until at some stage the candidate is rejected or all the stages are passed. The word "boosted" means that the classifiers at every stage of the cascade are complex themselves and they are built out of basic classifiers using one of four different `boosting` techniques (weighted voting). Currently Discrete Adaboost, Real Adaboost, Gentle Adaboost and Logitboost are supported. The basic classifiers are decision-tree classifiers with at least 2 leaves. Haar-like features are the input to the basic classifiers, and are calculated as described below. The current algorithm uses the following Haar-like features:



The feature used in a particular classifier is specified by its shape (1a, 2b etc.), position within the region of interest and the scale (this scale is not the same as the scale used at the detection stage, though these two scales are multiplied). For example, in the case of the third line feature (2c) the response is calculated as the difference between the sum of image pixels under the rectangle covering the whole feature (including the two white stripes and the black stripe in the middle) and the sum of the image pixels under the black stripe multiplied by 3 in order to compensate for the differences in the size of areas. The sums of pixel values over rectangular regions are calculated rapidly using integral images.