Robot car platform using a 3D camera sensor

HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Bachelor's thesis

Degree Programme in Electrical and Automation Engineering

Valkeakoski, Summer 2017

Hai Pham

HÄMEEN AMMATTIKORKEAKOULU
HÄME UNIVERSITY OF APPLIED SCIENCES

Electrical and Automation engineering
Valkeakoski

Author	Hai Pham	**Year** 2018
Subject	Robot car platform using 3D camera sensor	
Supervisor(s)	Katariina Penttilä, Antti Aimo	

ABSTRACT

In this thesis, a robot car platform was designed, which had the ability to visualize the 3D imaging data by analysing the signal from the 3D camera sensor. With the development in automobile engineering, it is necessary to produce such a robot car which can not only record geographic data but can also produce higher achievements in terms of data acquisition for data science.

This robot car platform was to be equipped with a 3D camera sensor at the front of the model for collecting data. In addition, to test the program, a robot car was built with two layers. The top layer was designed to store the controller parts including microcontrollers and motor drivers and the bottom one contained the power supply for the system.

Keywords Robot car platform, 3D camera sensor, microcontroller, data acquisition.

Pages 37 pages

CONTENTS

1 INTRODUCTION

The theoretical principle of this thesis project was mainly based on mobile robotics, specifically on the methods of controlling a robot's movement and data visualization. Robotics is an industry which is a combination of three widely different fields: mechanical engineering, electrical engineering and software engineering. The aim of this modern industry is to exploit machines to substitute human labour in a variety of working environments. A robot is not only able to imitate human activities but can also execute multi-tasking, following the path of programming (Lima & Ribeiro, 2002). Mobile robotics produces mobile robots, which are robots that have the ability to travel around their working environment. A mobile robot is programmed and controlled by software and receives feedback signals from the environment by using some kind of sensors such as an encoder, an imaging sensor, an ultrasonic sensor, etc. Moreover, artificial intelligence can be applied into mobile robotics, which can support the robot to navigate its position. (mobile robotics, n.d.)

Mobile robotics has a long period of development. In 1950, Grey Walter (UK, University of Bristol) developed a three wheeled mobile robot with a light sensor, touch sensor, propulsion motor, steering motor, and computer (Lima & Ribeiro, 2002). After that, in 1969, Shakey who worked in Stanford University, made the first mobile robot using machine vision for controlling. His robot could do multiple tasks such as: recognizing objects, finding the route to the targets and performing some interacting activities to obstacles. From 1993 to 1994, The CMU Field Robotics Center (FRC) created a mobile robot named Dante II, which explored a volcano in Alaska. The Dante II has a tether cable attached on the rim, which supports it to move downward to sheer crater wall so that it can collect and analyse high temperature gases (Lima & Ribeiro, 2002). In 2003, NASA first launched a Mars exploration rover named Curiosity and it arrived in Mars in 2012. The most significant thing of this rover was that it has a radioisotope power system that generated electricity from the heat of plutonium's radioactive decay. Figure 1 shows the image of Curiosity (Greicius, n.d.).

Figure 1. NASA's Curiosity Mars rover (Greicius, n.d.)

There are some challenges in controlling a mobile robot:

- Mechanical challenges: the robot needs an efficient mechanical system to move around the working environment. It is necessary to find the best solution that can reduce friction between components and consider the torque of motors to load the robot's weight.
- Electrical challenges: the power consumption of the robot must guarantee that the power loss will be decreased as much as it can. As the results, the robot can work with long life span and avoid being shut down suddenly. Besides, all of connections in electrical wiring must have protective switch to prevent devices being broken when executing multiple tasks.
- Programming challenges: the software controlling the robot must have logical algorithms so that the robot can analyse the input signal correctly and giving the precise results. Furthermore, the programs must be also executed in real-time without any delay.

2 THEORY

2.1 Embedded system

An embedded system is an integrated system. It is considered as a computer which is a combination of a mechanical or an electrical system (Barr & Massa , 1999). The main purpose of this system is to process the input and generate output. In the memory of an embedded system, an algorithm is built and stored to connect between inputs and outputs (Barrett, 2009). Figure 2 shows the architecture of an embedded system.

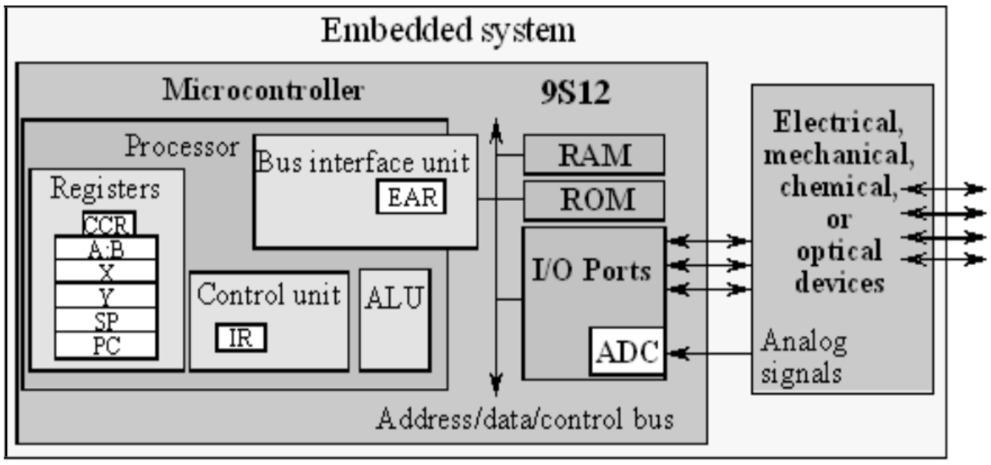


Figure 2. Embedded system (Jonathan, 2000)

To be a perfect embedded system, designers must consider these characteristics: reducing power lost and heat, reducing the dimensions without losing efficient, ability to work in multiscale of environment, and reasonable price for each unit. As the results, there is a limitation in processing resource and programming. To improve this limitation, the hardware must be built with an intelligence mechanism. The sensors and the network are also optimized to support managing resources. (Santiago, 2017).

2.2 **Programming background**

2.2.1 Python

Python is a programming language which has three main characteristics: interpreted, object-oriented and high-level programming. The data structure of Python is built at a high level, which includes abilities of typing and binding dynamically. One of Python's advantages is that the syntax is not only simple and easy to learn but also readable. As a result, the cost of programming and maintenance can be reduced significantly. Developers can program as module programming or code reuse thanks to Python’s packages and modules. In addition, the source code and library of Python is fully opened and free of charge, which is attractive to the programmers for developing in variety of platforms. (Blurb, n.d.)

Regularly, Python is one of the best choices for software developers because of its productivity. In comparision to C++, there is no compilation step so that it is fast for debugging, testing and editing code. When the interpreter discovers an error, it raises an exception.

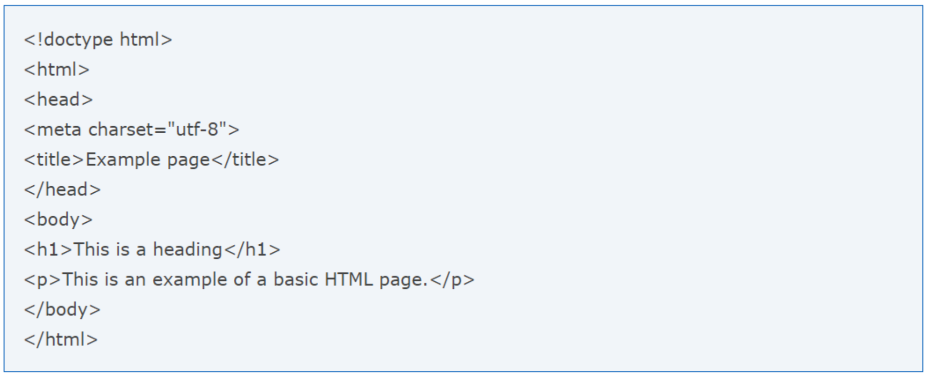
If there is no caught exception, a stack trace is printed by interpreter. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is created in Python, which the developer can also debug a program in an effective way by printing statements to the source. (Blurb, n.d.)

2.2.2 HTML

HTML is a programming language which is used for creating a website. Users can use the internet and a browser to access the website created by HTML. Similarly to Python, it is easy to code and powerful for web developers. HTML is designed and maintained by a company called "W3C" so that it is updated and upgraded regularly to adapt to the requirements and demands of Internet users. (Shannon, 2012)

HTML stands for "Hyper Text Markup Language". All the elements are defined the syntax and placement to build the structure of a website. There are some tags which provide the instruction for browsers to display the contents. HTML tags are also used to make some external links to other website or document in both local network and broadcast network. (Niederst, 2001)

The main function of tags in HTML is to illustrate the elements. Each HTML tag has its own attribute and it is typed between couple of angle bracket "<>". The content inside those brackets is not shown in the browser. The name of each tag usually expresses the attribute of it. (Niederst, 2001). Figure 3 shows an example of HTML.

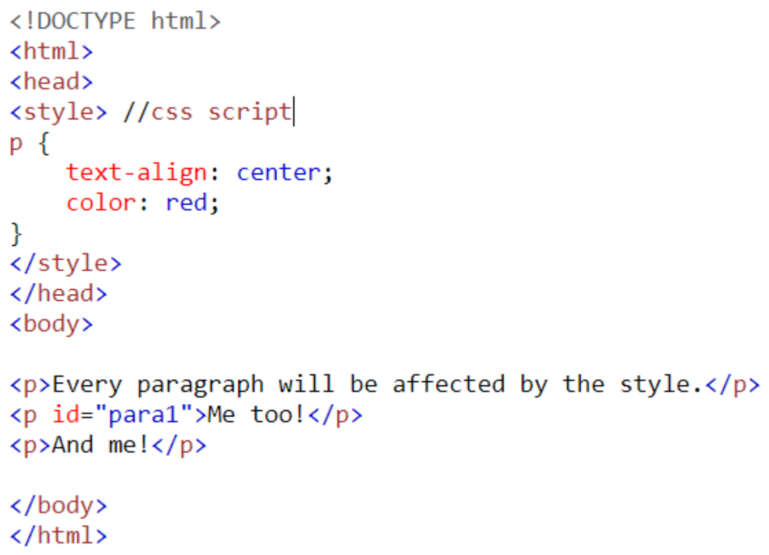


```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Example page</title>
</head>
<body>
<h1>This is a heading</h1>
<p>This is an example of a basic HTML page.</p>
</body>
</html>
```

Figure 3. Example of HTML (Jargon, 2017)

2.2.3 CSS

Cascading Style Sheets (CSS) is a style sheet language which has the ability to illustrate the presentation of a web page. The programmer can adjust the color, layout and font of the web page. Thanks to CSS, the web page can be responsive to many devices including large screens, small screens or printers. CSS can be coded with any XML-based markup language. With the independence to HTML, it is simple to remain sites, use style sheets across multiple environments with CSS. (web design, n.d.). Figure 4 shows an example of CSS code.



```
<!DOCTYPE html>
<html>
<head>
<style> //css script
p {
    text-align: center;
    color: red;
}
</style>
</head>
<body>

<p>Every paragraph will be affected by the style.</p>
<p id="para1">Me too!</p>
<p>And me!</p>

</body>
</html>
```

Figure 4. Example of CSS (CSS, n.d.)

2.2.4 Javascript

JavaScript is an object-oriented programming language which is lightweight interpreted. JavaScript provide executable contents which is expressed in the web page. Therefore, the website becomes more dynamically and controllable. (Flanagan, 1998) Besides, JavaScript is also a scripting language which contains a standard library such as Array, Date, and Math, and elements such as operators, control structures, and statements. (Javascript introduction, n.d.)

With client-side JavaScript, user can control document appearance and content. This language has “write()” method to write the arbitrary HTML as document parsed by browser. Moreover, in JavaScript, developers can also specify color, background or hypertext link in web page. In addition, the behaviour of browser can be controlled by client-side JavaScript. The object named “Window” allows users to display simple messages to get input from other users. Client-side JavaScript is also interacted with HTML form. (Flanagan, 1998)

Besides, Server-side JavaScript provides an alternative to GCI script. It is embedded within HTML, which creates executable scripts to be intermixed with web content. When receiving a request from clients, the server run the scripts including document before returning the results to the one requested. To improve the speed of sending request from client, it is necessary to precompiled HTML files contained server-side JavaScript to a binary form. (Flanagan, 1998)

2.3 **Electronics**

2.3.1 Microprocessor

A microprocessor plays an important role in a computer. It controls the computer so that it can execute ALU (Arithmetic Logical Unit) operations. The microprocessor has the ability to connect to other devices by some communications. There are three main parts which construct a microprocessor: ALU, register array, and a control unit. ALU controls arithmetical and logical operations on the data from the memory or input devices. Register array is a group of registers which is identified by letters and accumulator. The control unit manages the flow of data and instructions within the computer. The microprocessor follows a sequence: fetch, decode, and execute. (Microprocessor overview, n.d.)

Firstly, memory stores the instructions in a sequential order. After that, those instruction are fetched, decoded and executed until STOP instruction is reached. Then, microprocessor transfers the output value which is binary form the output port. When processing data, the register can save the data temporally and ALU computes with variety of functions. (Microprocessor overview, n.d.)

2.3.2 Microcontroller

A microcontroller is an integrated circuit which is designed to do some tasks or provide some applications (microcontroller, n.d.). Microcontroller has three main parts: ALU, memory and communication. The ALU's task is to perform arithmetic and other manipulations on data. ALU also must include read-only memory (ROM) and be non-volatile so that the program is remained even the power is off. The memory data of microcontroller can be random-access memory (RAM). Besides, the communication of microcontroller contains input and output ports and they can receive digital or analog signal. Figure 5 shows the structure of a microcontroller. (Volosyak, 2016).

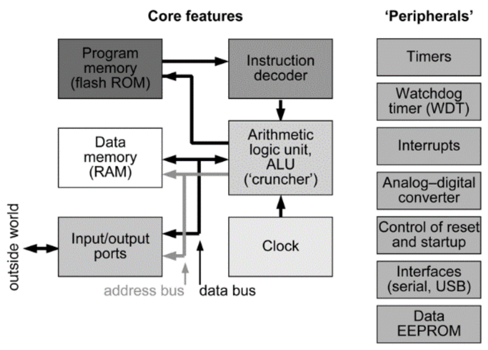


Figure 5. Structure of a microcontroller (Volosyak, 2016)

2.3.3 3D imaging

With two eyes set apart, a human can sense the depth and vertical – horizontal information. Because the eyes are in two different position, they allow the human to view an object with different dimensions. After that, the vision from the eye is transferred to the brain. The dimensions that humans recognize in their vision from the brain combining disparate images into a whole is a phenomenon called parallax. Similarly, each time when 3D shooting, two lenses are used to capture different images which are offset to each other. As a result, 3D images have twice as much information as 2D ones. The images are not only modified to display but also remained full of data. (How 3D Imaging works, n.d.)

There are many applications of 3D imaging such as analysing, measuring, and positioning objects. 3D imaging can be collected by using active or passive methods. Active methods include time-of-flight, structured light, and interferometry, which demand filming environment with a high degree of controlling. Passive methods contain depth from focus and light field. In snapshot-based methods, the difference between two snapshots captured at the same time is used to calculate the distance to objects. It can be achieved by moving a single camera and using two cameras with identical specifications is more efficient. By contrast, time-of-flight method encoded 3D data into each pixel by measuring the time period between sending light beam and receiving it. Another effective method for producing 3D shape data is laser triangulation. In laser triangulation, a single camera is used to define height variations from laser patterns projected onto the surface of an object, then observes how those patterns move when viewed from an angle with a camera. 3D imaging can also be implemented in other ways depending on the project and technology available. (How 3D Imaging works, n.d.)

3 METHODOLOGY

3.1 **Architecture**

A system architecture or systems architecture is the conceptual model that defines the structure, behaviour, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviours of the system.

A system architecture can comprise system components, the expand systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages (ADLs). Figure 6 illustrates the hardware architecture of the system.

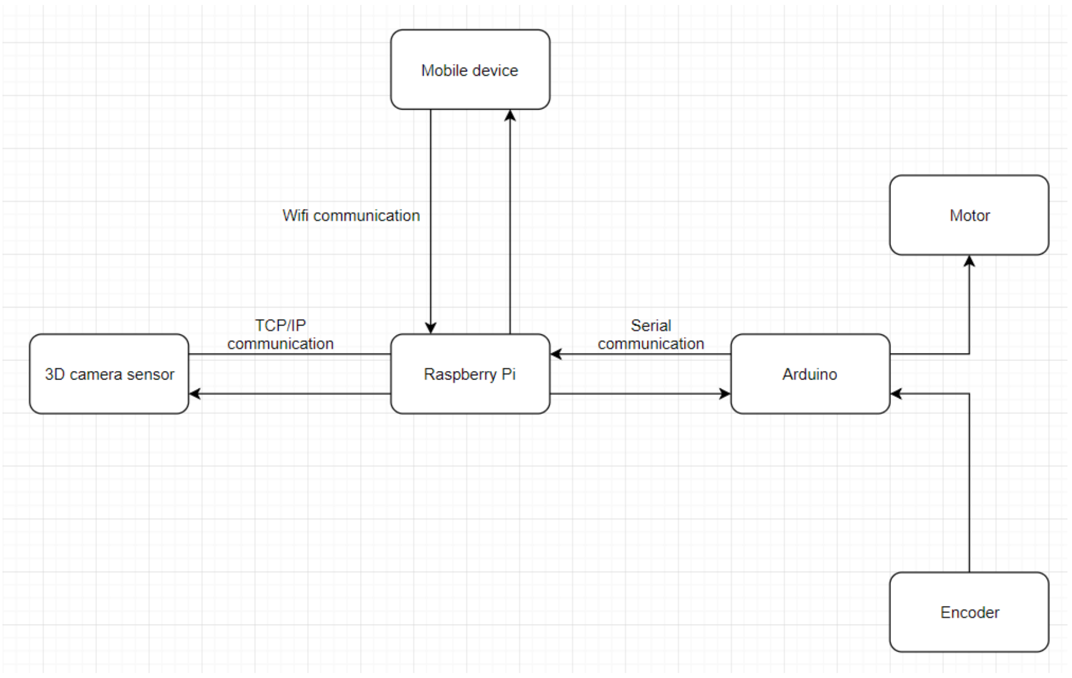


Figure 6. Architecture of hardware system

According to Figure 6, data will be collected by the 3D camera sensor before being processed by Raspberry Pi. This computer will execute some programs to visualize the image that the 3D camera sensor has already collected. To control the movements of the car, user can use the user interface whose function is to send the commands to microcontroller - Arduino to adjust the speed and direction of the motors. The Arduino can also read the encoder value from hall sensor attached to the motor, which is used for angle calculation in data visualization.

3.2 Testing model

The testing model was designed as a two-floor car with the first floor as a battery storage and the second floor as a controller box. Figure 7 shows the image of the testing model.

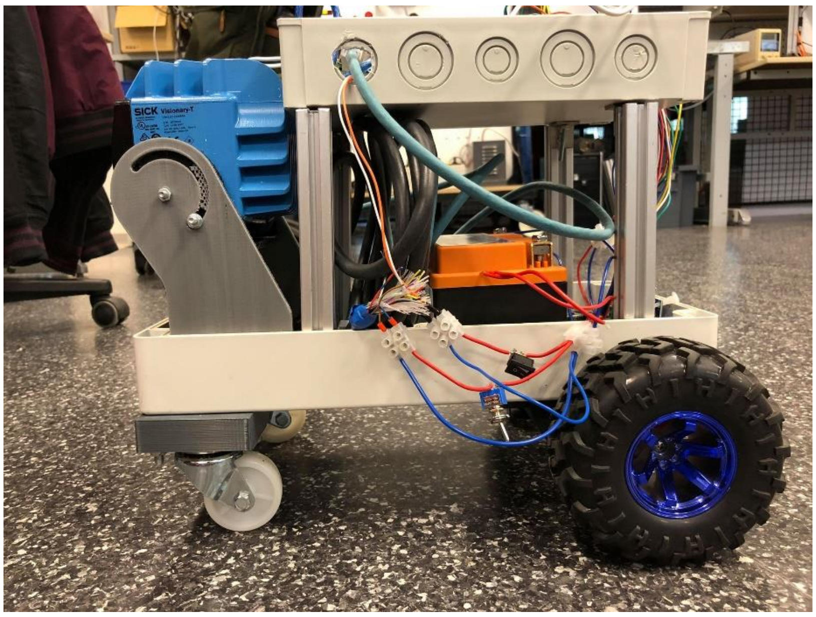


Figure 7. Testing model

The model had two rotary wheels in the front and two normal wheels in the rear. With the rotary wheels, the car could turn by running the two rear wheels in the opposite direction. As a result, the equation (1) shows how to calculate the position of the car when moving which is also illustrated in Figure 8:

$$\text{Position} = \begin{matrix} \cos(\varphi) & -\sin(\varphi) & 0 & X \\ \sin(\varphi) & \cos(\varphi) & 0 & Y \\ 0 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 \end{matrix} \tag{1}$$

+ φ is the angle of the car when rotating around Z axis (deg)
+ X and Y is the position in x-y coordination (mm)
+ Z car lower floor height (mm)

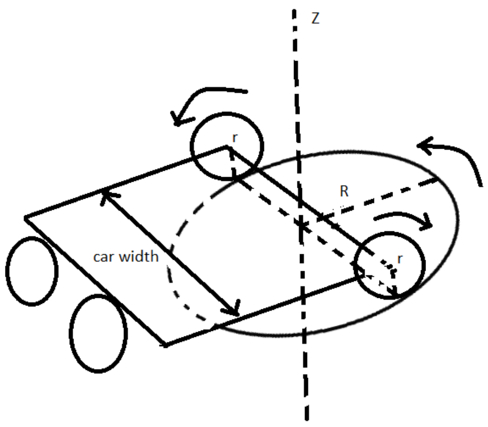


Figure 8. Rotation of the car

$$\varphi = \frac{2*r*Rw}{R} * 360(deg) \qquad (2)$$

$$displacement = Rw * 2 * \pi * r \text{ (mm)}$$

+ r : diameter of rear wheel (mm)
+ R : distance between two rear wheels (mm)
+ Rw: revolution of each rear wheel (round)

3.3 **Electrical design**

Electrical design is a procedure which contains a variety of work such as: planning, creating, testing, or supervising the development and installation of electrical equipment, including lighting equipment, power systems, power distribution, fire and life safety systems, electronic components, and voice and data communications infrastructure. In this embedded system, we designed the electrical system into three components: the power supply, the controller and the devices.

3.3.1 Controllers

a. Raspberry Pi

The Raspberry Pi is a small computer which is designed in a simple architecture. It has ability to connect to other peripheral devices such as screen monitor, mouse or keyboard. Users are easy to study how to code and develop software with raspberry pi because it supports variety of languages including Python, C++ or JavaScript. It can also do other activities which a normal computer is able to do such as Internet browsing, playing video, etc. (What is a Raspberry Pi, n.d.)

However, there are some limitation when using Raspberry pi. Firstly, the memory of Raspberry Pi is only 1GB which is not large enough for developers who would like to develop some application required big data. Besides, the processor is not effective enough to update some modern software. As the result, some program cannot run in full speed as it should. (McManus & Cook, 2013). Figure 9 shows the image and datasheet of a Raspberry Pi.

Processor:	Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
Memory:	1GB LPDDR2 SDRAM
Connectivity:	■ 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE ■ Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps) ■ 4 × USB 2.0 ports
Access:	Extended 40-pin GPIO header
Video & sound:	■ 1 × full size HDMI ■ MIPI DSI display port ■ MIPI CSI camera port ■ 4 pole stereo output and composite video port
Multimedia:	H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics
SD card support:	Micro SD format for loading operating system and data storage
Input power:	■ 5V/2.5A DC via micro USB connector ■ 5V DC via GPIO header ■ Power over Ethernet (PoE)–enabled (requires separate PoE HAT)
Environment:	Operating temperature, 0–50°C

Figure 9. Raspberry Pi model 3 B+ (Product briefs, n.d.)

b. Arduino

Arduino is a microcontroller board which is designed as a platform with simple hardware and software. Arduino can implement multiple tasks: reading inputs (button state, analog signal), turning to output (activating motor, LED) or releasing data online. Users can use Arduino IDE to start programming and uploading the code into the microcontroller with interactive interface. (Introduction, n.d.).

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13
Length	101.52 mm
Width	53.3 mm
Weight	37 g

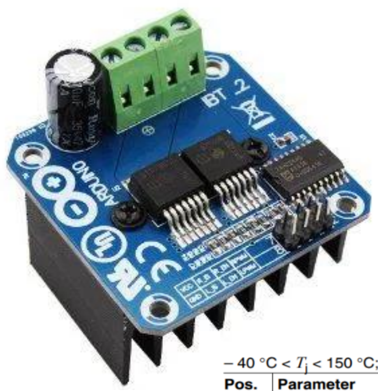
Figure 10. Arduino mega 2560 Rev3 (Arduino-mega-2560-rev3, n.d.)

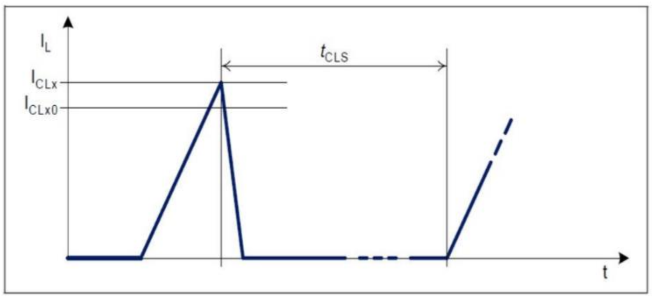
c. Motor driver BTS7960 43A

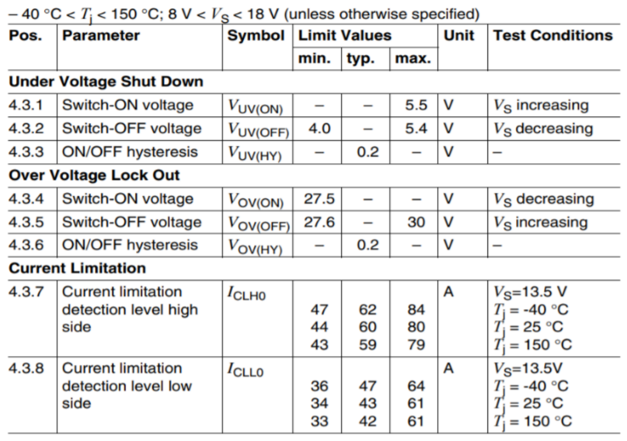
A motor driver works as a current amplifier which has ability to turn from a low-current control signal to a high-current signal that can drive a motor. The categories of motor are classified depending on the amount of voltage to be operated, the value of output current, percentage of power lost, packed methods and output quantities. (Motor driver, n.d.)

The BTS 7960 is a fully integrated high current driver for motor. The operating voltage is 24V at maximum and continuous current is 43A, the range of PWM frequency limitation is 25 kHz. The most effective thing in this driver is that it is customized with protection characteristics:

- Limit low voltage: the circuit will shut down when voltage is below 5.4 volts.
- Overtemperature protection: there is an integrated temperature switch which can shut down output when temperature reach to the limit (200^{o}C). This advantage not only protect the driver but also the actuators connected to the output ports.
- Current limitation: if the current approaches to the limit current (Iclx), one switch is turned off and the other switch is activated for a certain time (Tcls). (Rawashdeh, 2014)





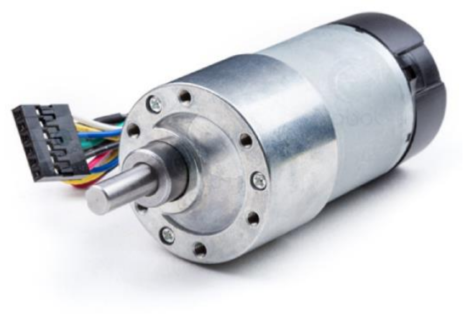

– 40 °C < T_j < 150 °C; 8 V < V_S < 18 V (unless otherwise specified)

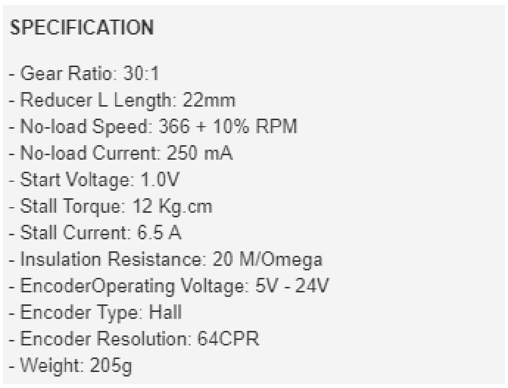
Pos.	Parameter	Symbol	Limit Values			Unit	Test Conditions
			min.	typ.	max.		
Under Voltage Shut Down							
4.3.1	Switch-ON voltage	$V_{UV(ON)}$	–	–	5.5	V	V_S increasing
4.3.2	Switch-OFF voltage	$V_{UV(OFF)}$	4.0	–	5.4	V	V_S decreasing
4.3.3	ON/OFF hysteresis	$V_{UV(HY)}$	–	0.2	–	V	–
Over Voltage Lock Out							
4.3.4	Switch-ON voltage	$V_{OV(ON)}$	27.5	–	–	V	V_S decreasing
4.3.5	Switch-OFF voltage	$V_{OV(OFF)}$	27.6	–	30	V	V_S increasing
4.3.6	ON/OFF hysteresis	$V_{OV(HY)}$	–	0.2	–	V	–
Current Limitation							
4.3.7	Current limitation detection level high side	I_{CLH0}	47 44 43	62 60 59	84 80 79	A	V_S=13.5 V T_j = -40 °C T_j = 25 °C T_j = 150 °C
4.3.8	Current limitation detection level low side	I_{CLL0}	36 34 33	47 43 42	64 61 61	A	V_S=13.5V T_j = -40 °C T_j = 25 °C T_j = 150 °C

Figure 11. BTS7960 specifications (Rawashdeh, 2014)

3.3.2 Motor with encoder

In this project I used a 12V motor which was assembled with a quadrature encoder. The motor has a 30:1 gear ratio and has 12 kg.cm torque value. The encoder of the motor is a hall sensor which has two channels (A and B) to measure the value of revolution of the motor. The total resolution of the encoder is 64CPR. Moreover, the motor can rotate up to 366 rpm in no load situation. (product, n.d.). Figure 12 shows the image and the specifications of the motor.





SPECIFICATION

- Gear Ratio: 30:1
- Reducer L Length: 22mm
- No-load Speed: 366 + 10% RPM
- No-load Current: 250 mA
- Start Voltage: 1.0V
- Stall Torque: 12 Kg.cm
- Stall Current: 6.5 A
- Insulation Resistance: 20 M/Omega
- EncoderOperating Voltage: 5V - 24V
- Encoder Type: Hall
- Encoder Resolution: 64CPR
- Weight: 205g

Figure 12. 12V motor with hall quadrature sensor encoder (product, n.d.)

The encoder of the motor has two channels A and B to indicate the revolution of the motor. These channels' phase is apart 90 degrees. If the motor run in counter clockwise, the channel A is 90 degrees faster than channel B and the opposite result happens when regulating counter clockwise. As a result, the user can define the direction of the motor by sensing change of pulse between channel A and B. Moreover, by counting the time of HIGH state and LOW state for each channel, the users can retrieve that data to calculate the rotation of motors. (quadrature encoder, n.d.)

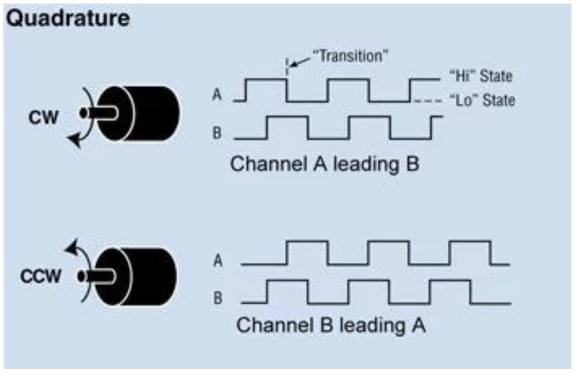


Figure 13. How quadrature sensor works (quadrature encoder, n.d.)

3.3.3 SICK 3D camera sensor

In this project, I used the visionary-T 3D vision sensor from SICK. The main function of this sensor is providing 3D snapshots in real-time. Based on time-of-flight measurement, the sensor can give depth data and the raw data is processed as some parameters depending on the purpose of usage. Visionary-T has high performance and visualization which is a good option for developing in robotics, vehicle and so on. Figure 14 shows the image of the sensor and Figure 15 shows specifications of one. (Visionary T, n.d.).

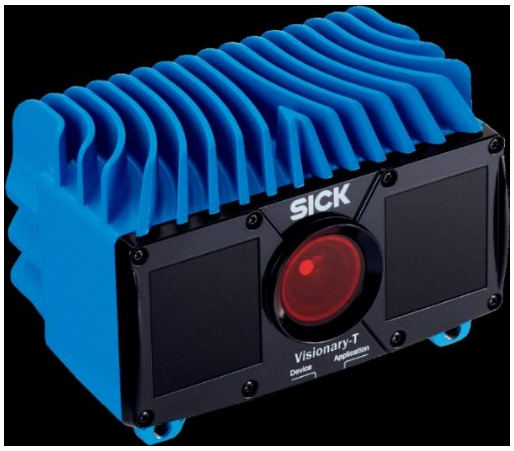
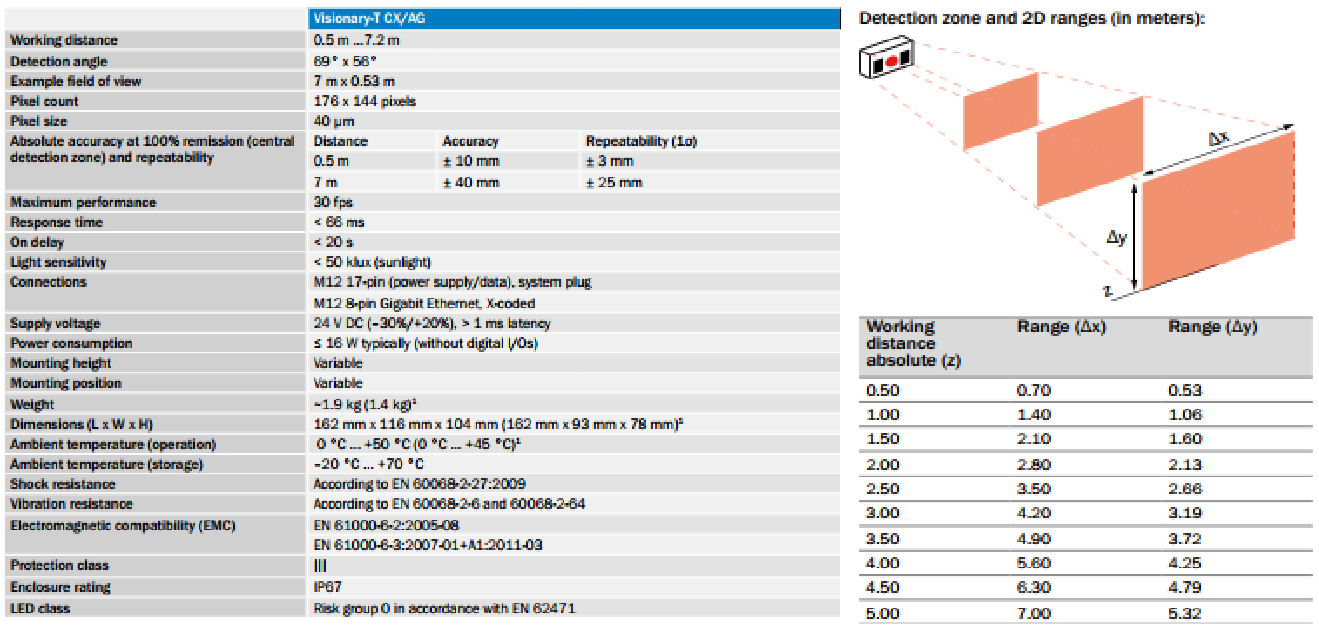


Figure 14. SICK visionary T V3S110-1AAAAAA (Visionary T, n.d.)



	Visionary-T CX/AG		
Working distance	0.5 m ...7.2 m		
Detection angle	69° x 56°		
Example field of view	7 m x 0.53 m		
Pixel count	176 x 144 pixels		
Pixel size	40 µm		
Absolute accuracy at 100% remission (central detection zone) and repeatability	Distance	Accuracy	Repeatability (1σ)
	0.5 m	± 10 mm	± 3 mm
	7 m	± 40 mm	± 25 mm
Maximum performance	30 fps		
Response time	< 66 ms		
On delay	< 20 s		
Light sensitivity	< 50 klux (sunlight)		
Connections	M12 17-pin (power supply/data), system plug		
	M12 8-pin Gigabit Ethernet, X-coded		
Supply voltage	24 V DC (-30%/+20%), > 1 ms latency		
Power consumption	≤ 16 W typically (without digital I/Os)		
Mounting height	Variable		
Mounting position	Variable		
Weight	~1.9 kg (1.4 kg)^1		
Dimensions (L x W x H)	162 mm x 116 mm x 104 mm (162 mm x 93 mm x 78 mm)^1		
Ambient temperature (operation)	0 °C ... +50 °C (0 °C ... +45 °C)^1		
Ambient temperature (storage)	-20 °C ... +70 °C		
Shock resistance	According to EN 60068-2-27:2009		
Vibration resistance	According to EN 60068-2-6 and 60068-2-64		
Electromagnetic compatibility (EMC)	EN 61000-6-2:2005-08		
	EN 61000-6-3:2007-01+A1:2011-03		
Protection class	III		
Enclosure rating	IP67		
LED class	Risk group 0 in accordance with EN 62471		

Detection zone and 2D ranges (in meters):

Working distance absolute (z)	Range (Δx)	Range (Δy)
0.50	0.70	0.53
1.00	1.40	1.06
1.50	2.10	1.60
2.00	2.80	2.13
2.50	3.50	2.66
3.00	4.20	3.19
3.50	4.90	3.72
4.00	5.60	4.25
4.50	6.30	4.79
5.00	7.00	5.32

Figure 15. Sick visionary T specification (Visionary T, n.d.)

The visionary T works followed principle of time-of-flight. The 3D camera sensor extracts beam of light to the object and receives the reflected beam. After that, it calculates the time of the process and return the value of distance. Figure 16 shows the process of send and receive the signal from the sensor.

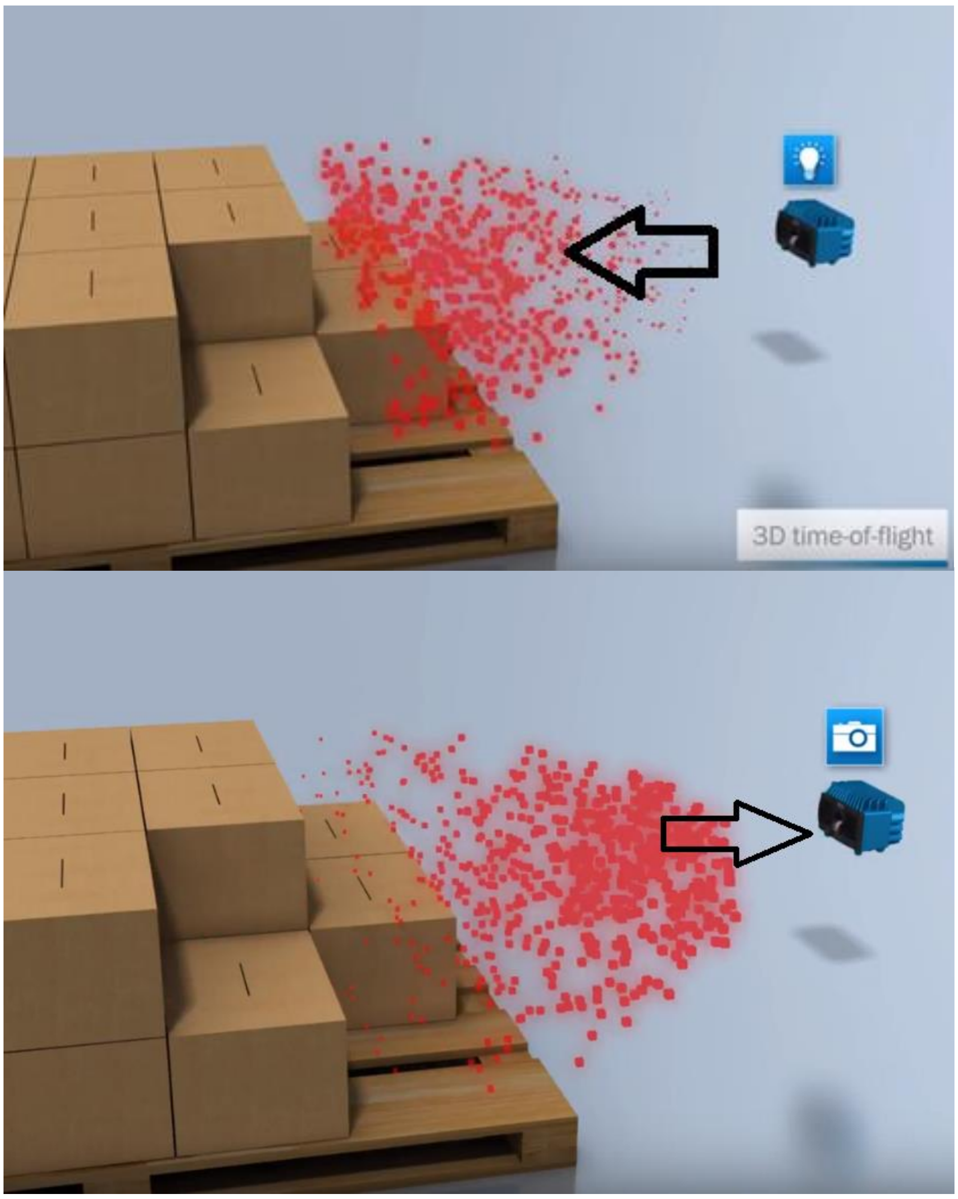


Figure 16. Sending and receiving signal of Visionary – T (Intelligence, n.d.)

When sending the signal, the sensor creates multiple frames which are divided into small pixels. The size of frame is 176px*144px. For each frame, the distance from the plane of sensor to each particle of object will be calculated by the following equation:

$$\text{Distance plane to object} = \frac{Distance(ToF)}{\cos(\alpha)} \tag{3}$$

+ α: the angle between light beam and the frame
+ OH: distance of sensor plane to the object
+ OP: distance ToF

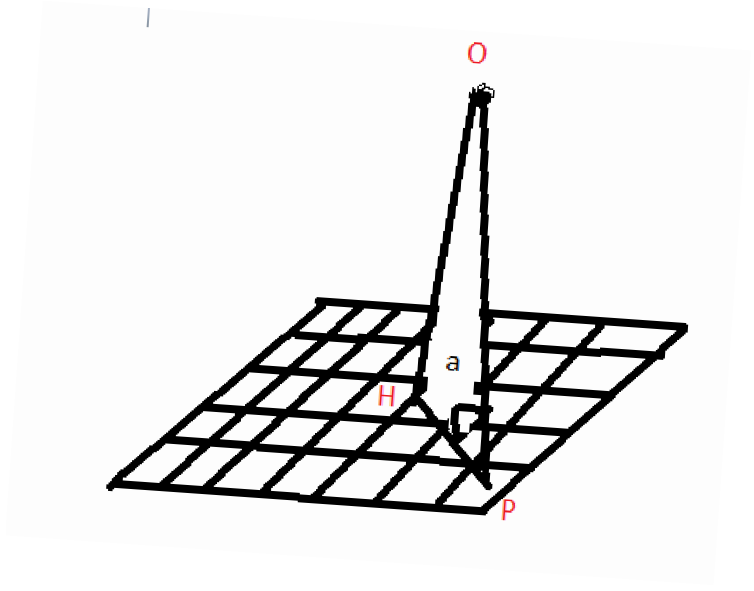


Figure 17. Projection of points in frame

In processing data of Visionary-T, there are two types of connection: streaming and control, which is shown in Figure 18. Streaming data allow device to send out cyclic data as known as blobs (binary large objects). These blobs contain distance, intensity and confidence maps as well as camera parameters. For the control communication, the sensor is able to read variable or execute some methods. (CID Visionary-T AG)

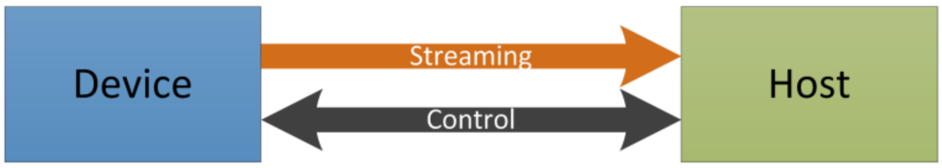


Figure 18. Visionary T communication (Communication interface description)

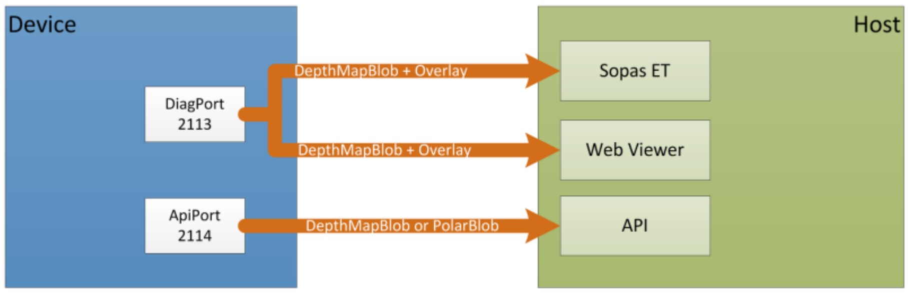


Figure 19. Streaming data (Communication interface description)

To execute some method of sensor, the user must program application which can send the binary telegram sequence to the sensor. Figure 20 shows the principle of binary telegram sequence.

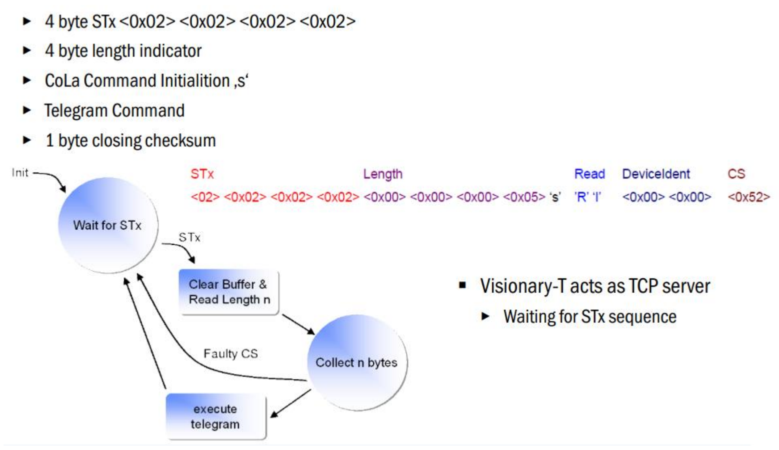
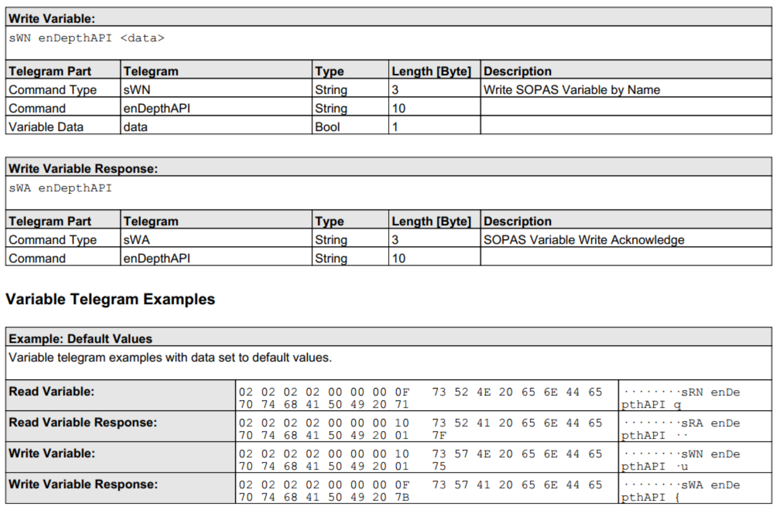

4 byte STx <0x02> <0x02> <0x02> <0x02>
4 byte length indicator
CoLa Command Initiation ‚s'
Telegram Command
1 byte closing checksum
Init
Wait for STx
STx
Clear Buffer & Read Length n
Collect n bytes
Faulty CS
execute telegram
STx
Length
Read
DeviceIdent
CS
<02> <0x02> <0x02> <0x02> <0x00> <0x00> <0x00> <0x05> 's' 'R' 'I' <0x00> <0x00> <0x52>
Visionary-T acts as TCP server
Waiting for STx sequence

Figure 20. Principle of binary telegram sequence (Communication interface description)



Write Variable:

```
sWN enDepthAPI <data>
```

Telegram Part	Telegram	Type	Length [Byte]	Description
Command Type	sWN	String	3	Write SOPAS Variable by Name
Command	enDepthAPI	String	10	
Variable Data	data	Bool	1	

Write Variable Response:

```
sWA enDepthAPI
```

Telegram Part	Telegram	Type	Length [Byte]	Description
Command Type	sWA	String	3	SOPAS Variable Write Acknowledge
Command	enDepthAPI	String	10	

Variable Telegram Examples

Example: Default Values

Variable telegram examples with data set to default values.

Read Variable:	`02 02 02 02 00 00 00 0F   73 52 4E 20 65 6E 44 65`
	`70 74 68 41 50 49 20 71`
		`········sRN enDe`
		`pthAPI q`

Read Variable Response:	`02 02 02 02 00 00 00 10   73 52 41 20 65 6E 44 65`
	`70 74 68 41 50 49 20 01   7F`
		`········sRA enDe`
		`pthAPI ··`

Write Variable:	`02 02 02 02 00 00 00 10   73 57 4E 20 65 6E 44 65`
	`70 74 68 41 50 49 20 01   75`
		`········sWN enDe`
		`pthAPI ·u`

Write Variable Response:	`02 02 02 02 00 00 00 0F   73 57 41 20 65 6E 44 65`
	`70 74 68 41 50 49 20 7B`
		`········sWA enDe`
		`pthAPI {`

Figure 21. Telegram example (Communication interface description)

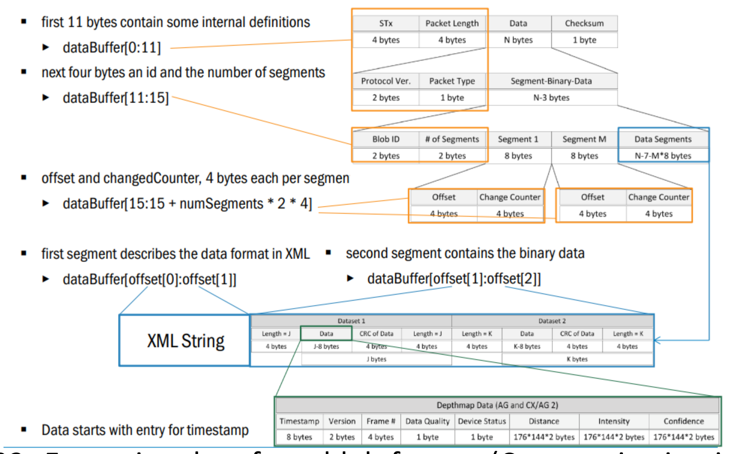


Figure 22. Extracting data from blob format (Communication interface description)

3.3.4 Power supply

Power supply plays an important role in an embedded system. If the power system cannot supply enough power, the car cannot work for a long time and other devices can be shut down immediately, which can cause some errors of those devices. In this platform, there are three solutions to build the power supply system:

- Controller, motor and sensor used the same power source: In this case, two 12V batteries are connected to create 24V source for the sensor. With this solution, the power supply of the system will be synchronous, which means all devices will be concurrency ON or OFF. There is one disadvantage when using same power supply for all devices is that each device (except 3D camera sensor) is required a voltage regulator (Raspberry pi, arduino – 5V and motor – 12V).
- Controller, motor and sensor used separated power source: In this case, one 12V battery is used for motors. The other 12V battery is used for the 3D camera sensor together with a boost converter step up power. In addition, the raspberry pi and Arduino are connected and use an extra 5V battery. In this solution, the safety standard is guaranteed because those devices have different power consumption and operating voltage level. As the result, there is no over voltage or current situation in this solution. In contrast, the weight of car is increased and those devices cannot work synchronously because of different battery's capacity.

Considering safe and power efficiency, the second solution is more suitable for the car platform. For the raspberry pi and Arduino, Figure 23 shows the image of battery with expansion board.

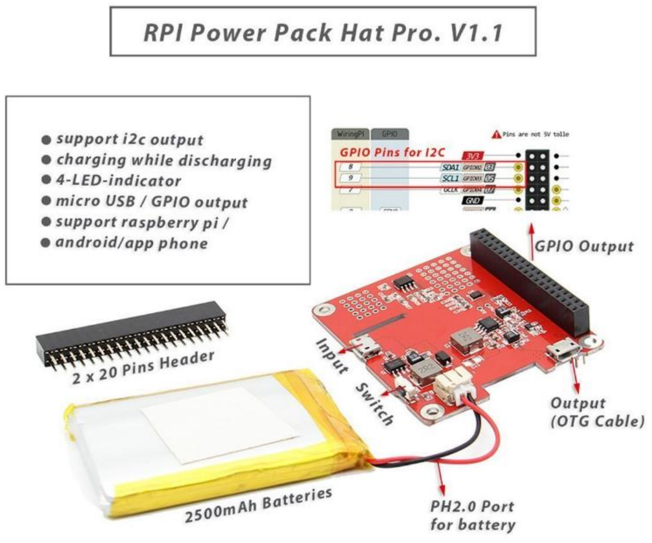
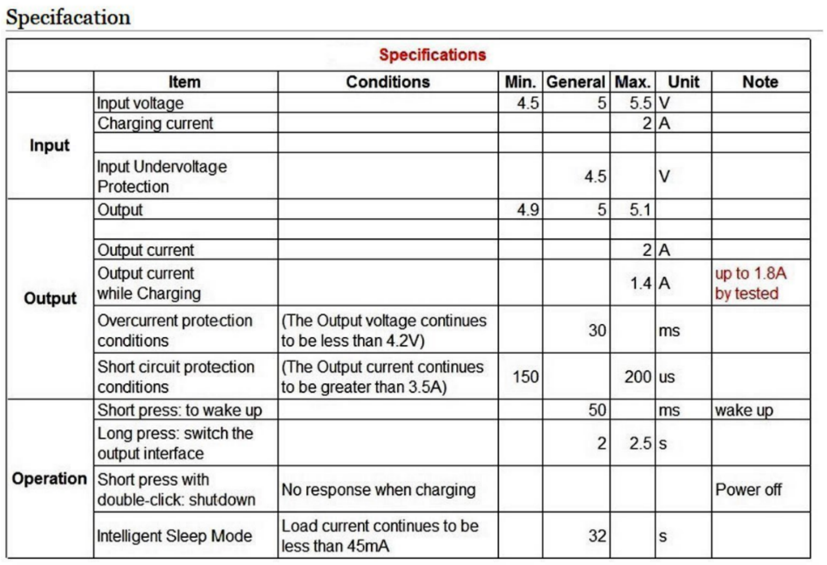


Figure 23. Maker Hawk raspberry pi expansion board power pack with Lithium Battery (B077M59T75, n.d.)


Specifacation

Specifications							
	Item	Conditions	Min.	General	Max.	Unit	Note
Input	Input voltage		4.5	5	5.5	V	
	Charging current				2	A	
	Input Undervoltage Protection			4.5		V	
Output	Output		4.9	5	5.1		
	Output current				2	A	
	Output current while Charging				1.4	A	up to 1.8A by tested
	Overcurrent protection conditions	(The Output voltage continues to be less than 4.2V)		30		ms	
	Short circuit protection conditions	(The Output current continues to be greater than 3.5A)	150		200	us	
Operation	Short press: to wake up			50		ms	wake up
	Long press: switch the output interface			2	2.5	s	
	Short press with double-click: shutdown	No response when charging					Power off
	Intelligent Sleep Mode	Load current continues to be less than 45mA		32		s	

Figure 24. Marker Hawk expansion board power pack specifications (B077M59T75, n.d.)

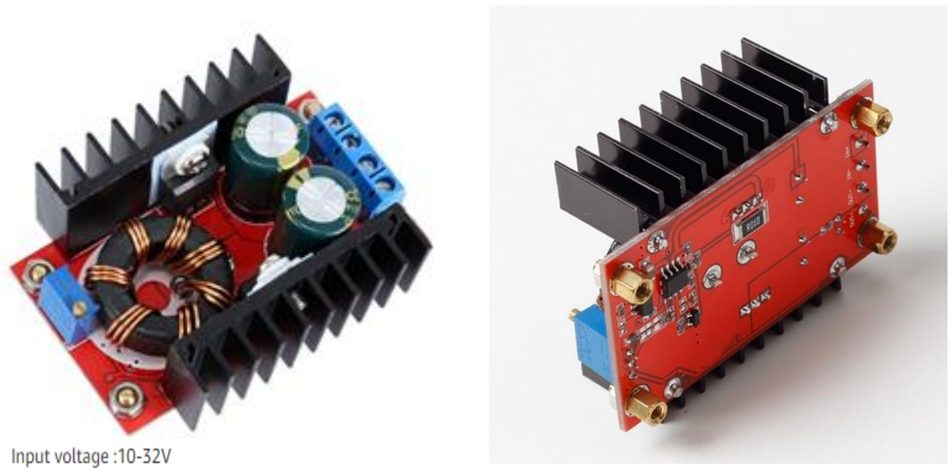
The advantage of using this battery kit is that it is easy to connect to the raspberry pi because of USB connection and the battery can be recharged. Moreover, it is customized with a switch so that users can shut down the raspberry pi immediately when hazardous accident happens.

Power supply for the motors is a 12V battery. Figure 25 shows the image and specification of the battery.

Voltage	12V
Power	60Wh (5.0Ah)
Start current	290A
Size	150mmx87mmx93mm
Weight	1 kg

Figure 25. Exide lithium – ion battery (Exide Litium akku 12V ELTZ14S, n.d.)

With the 3D camera sensor, the operating voltage is 24V so that a boost converter must be added to supply enough power for the sensor. Figure 26 shows the image and specifications of the boost converter.



Input voltage :10-32V
Output voltage: 12-35V (adjustable)
Output Current: 10A (MAX)
Input Current: 16A (MAX) (Please enhance heat dissipation if more than 10A)
Output power: natural cooling 100W (MAX), enhance heat dissipation 150W (MAX)
Easy to drive 65W 90W dual-core notebook.
Use a 12V battery drive 19V 3.42A notebook, the module temperature about 45°c
Conversion efficiency: 94% (measured at Input 16V, output 19V 2.5A)
Output Ripple: 2% (MAX) 20M-bandwidth
Operating Temperature: Industrial (-40°c to +85°c) (ambient temperature exceeds 40°c, lower power use, or to enhance heat dissipation)
Full load temperature rise: 45°c
No-load current: 25mA typical
Voltage regulation: ± 0.5%
Dynamic response speed: 5% 200uS

Figure 26. DC-DC boost converter 10-32V to 12-35V step up adjustable power supply 150W (B00HV43UOG, n.d.)

3.3.5 Electrical wiring

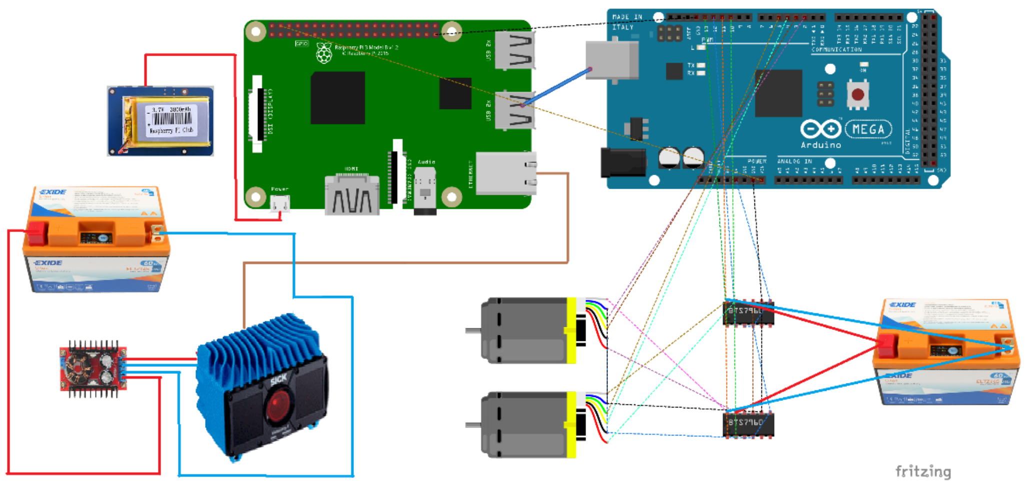


Figure 27. Device connection

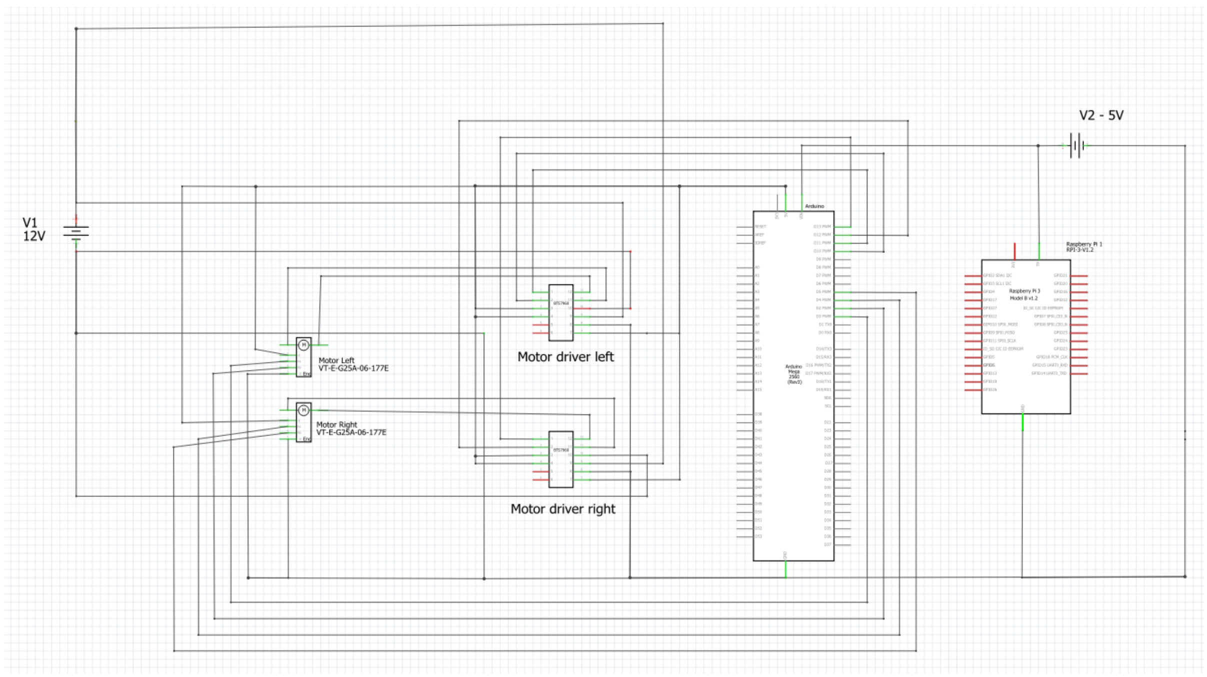


Figure 28. Electrical schematic

Table 1. IO controllers board connection

Battery	Raspberry Pi	Arduino	Motor driver left	Motor left	Motor driver right	Motor right
	USB port	Serial port				
		5V	Vcc		Vcc	
			R_EN		R_EN	
			L_EN		L_EN	
				4		4
		10	LPWM			
		11	RPWM			
		13			RPWM	
		12			LPWM	
5V	Micro USB					
12V			B+		B+	
-12V			B-		B-	
			M+	1		
			M-	2		
		3		5		
		5		6		
		GND	GND	3	GND	3
					M+	1
					M-	2
		2				5
		4				6

Table 2. 3D camera sensor connection

Battery	Raspberry Pi	DC-DC boost converter	3D camera sensor
12V		IN+	
12V-		IN-	
		OUT+	2
		OUT-	1
	Ethernet port		Gigabit ethernet

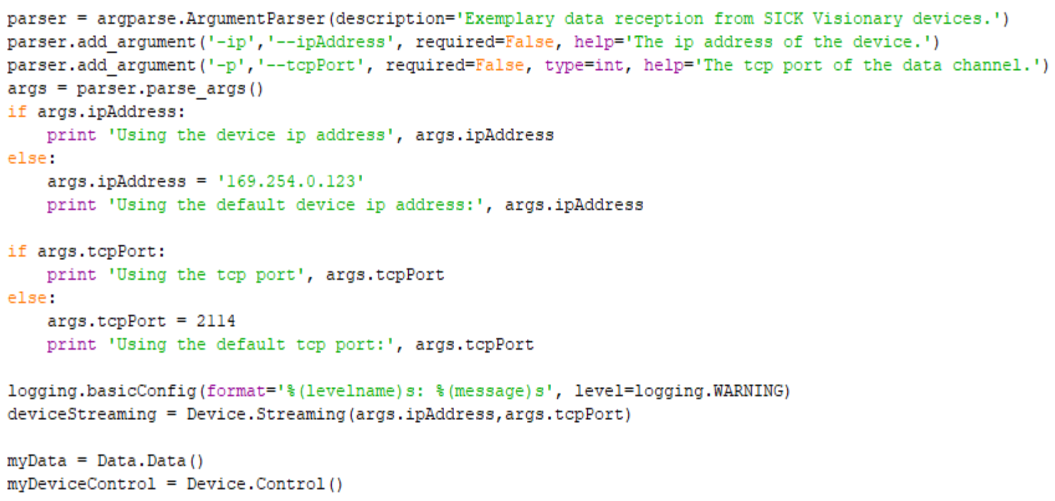
3.4 **Programming**

Programming is an important step when controlling the robot car platform. In this thesis, there are four main programs which not only communicate between users and the robot but also control devices of the platform.

3.4.1 Reading data from 3D camera sensor

The program which transfers data from the sensor to the raspberry pi names "readData.py". The program is divided into 3 parts: initializing socket connection, setting user level and reading data. For this program, there are two library which can be used: "Device.py" and "Data.py".

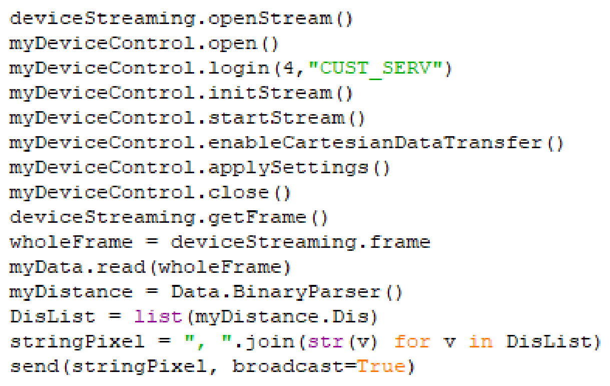
Figure 29 shows the code of initializing socket connection. Before running this program, the sensor must be configured in SoPAS to define the ip-address. In this case, the ip-address is "169.254.0.123".



```
parser = argparse.ArgumentParser(description='Exemplary data reception from SICK Visionary devices.')
parser.add_argument('-ip','--ipAddress', required=False, help='The ip address of the device.')
parser.add_argument('-p','--tcpPort', required=False, type=int, help='The tcp port of the data channel.')
args = parser.parse_args()
if args.ipAddress:
    print 'Using the device ip address', args.ipAddress
else:
    args.ipAddress = '169.254.0.123'
    print 'Using the default device ip address:', args.ipAddress

if args.tcpPort:
    print 'Using the tcp port', args.tcpPort
else:
    args.tcpPort = 2114
    print 'Using the default tcp port:', args.tcpPort

logging.basicConfig(format='%(levelname)s: %(message)s', level=logging.WARNING)
deviceStreaming = Device.Streaming(args.ipAddress,args.tcpPort)

myData = Data.Data()
myDeviceControl = Device.Control()
```

Figure 29. Initializing socket connection

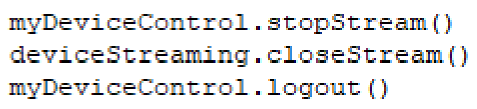
In the second step, the user need to login to the system and set the level of user. After that, the sensor can start collecting and streaming the data to the controller. Figure 30 illustrates the codes to log in and streaming the data.



```
deviceStreaming.openStream()
myDeviceControl.open()
myDeviceControl.login(4,"CUST_SERV")
myDeviceControl.initStream()
myDeviceControl.startStream()
myDeviceControl.enableCartesianDataTransfer()
myDeviceControl.applySettings()
myDeviceControl.close()
deviceStreaming.getFrame()
wholeFrame = deviceStreaming.frame
myData.read(wholeFrame)
myDistance = Data.BinaryParser()
DisList = list(myDistance.Dis)
stringPixel = ", ".join(str(v) for v in DisList)
send(stringPixel, broadcast=True)
```

Figure 30. Login and streaming data

If there are any interruptions when generating program, there are three commands which will be execute to close and log out of device. Figure 31 shows the exit step of the device.



```
myDeviceControl.stopStream()
deviceStreaming.closeStream()
myDeviceControl.logout()
```

Figure 31. Exit the program

3.4.2 Server

This robot car platform is controlled by mobile devices with wifi signal. There are some advantages of using wifi signal:

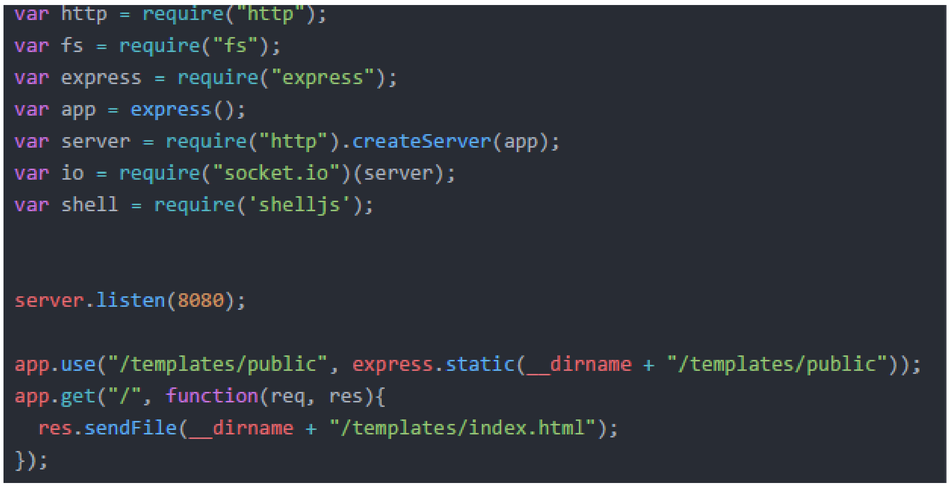
- The robot car can work from any location that it can receive the signal
- It is easy to set up and configure
- The wifi signal can be expanded depending on the space
- The user can access even moving around the working environment

However, there are some disadvantages of using wifi to control the robot:

- The speed of signal is slow
- The wifi signal is weakened when the robot passing through a wall
- High risk of security must be considered. (Shrestha, 2017)

To control the robot car by using wifi, a web server must be created. In this case, Node.js is a suitable and effective tool to do this task.

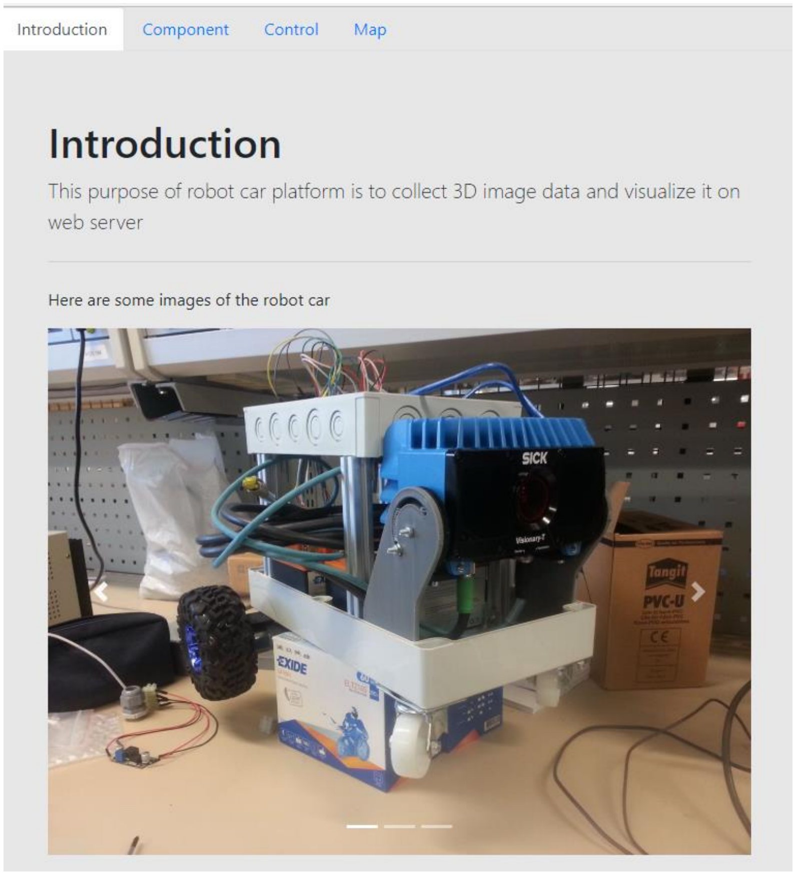
Node.js is a JavaScript runtime which is created on Chrome's V8 JavaScript engine. Node.js is efficient because of using an event-driven and non-blocking I/O model. Figure 32 shows the controlling server for the robot platform created by Node.js.



```
var http = require("http");
var fs = require("fs");
var express = require("express");
var app = express();
var server = require("http").createServer(app);
var io = require("socket.io")(server);
var shell = require('shelljs');


server.listen(8080);

app.use("/templates/public", express.static(__dirname + "/templates/public"));
app.get("/", function(req, res){
  res.sendFile(__dirname + "/templates/index.html");
});
```

Figure 32. Creating controlling server for robot platform

According to Figure 32, the server renders the file named “index.html” as user interface. To view the information of the robot platform and control it, users can access to the address “raspberrypi.local:8080” and the web browser will show an image as Figure 33.



Introduction Component Control Map

Introduction

This purpose of robot car platform is to collect 3D image data and visualize it on web server

Here are some images of the robot car

Figure 33. User interface

As Figure 23, there are four main tabs: Introduction, component, control and map. In the introduction tab, the purpose of the robot car is defined and the user can also see three picture of the platform. In the component tab, users are able to see all of the components which are assembled in the car. If the users click on each image of component, a product's link will be automatically opened and they can check all the datasheet. In addition, the control tab shows four button of direction to control the car. Figure 24 shows the image of component tab and control tab.

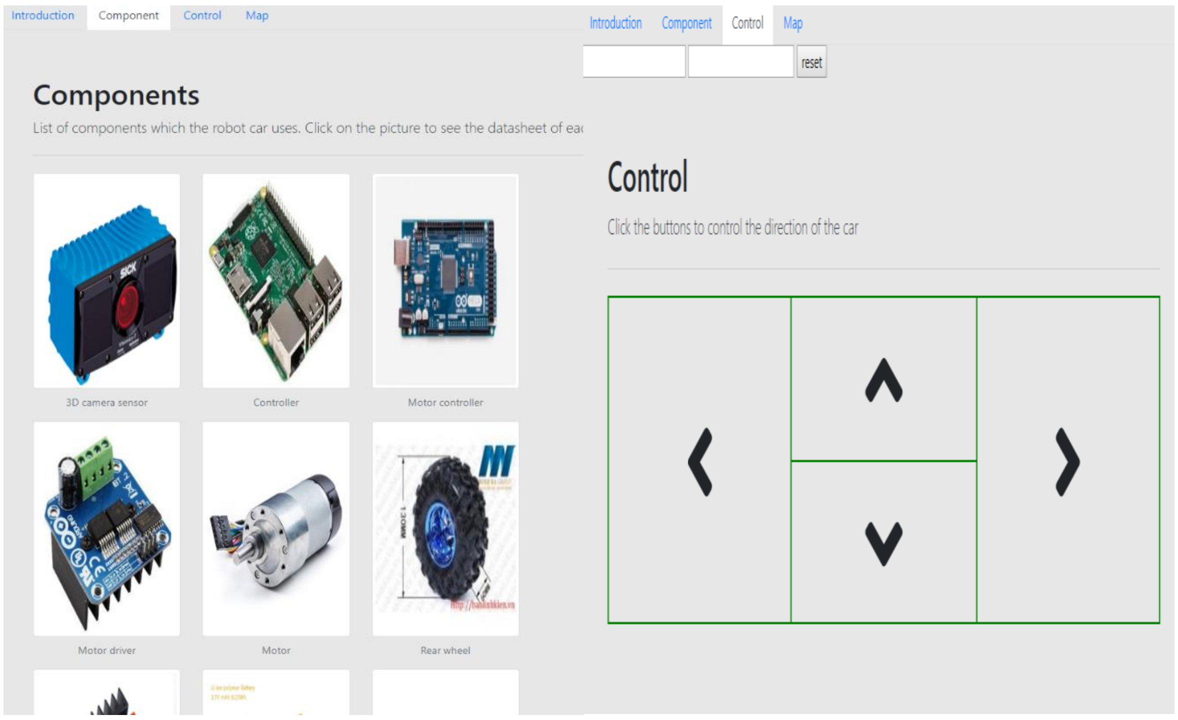
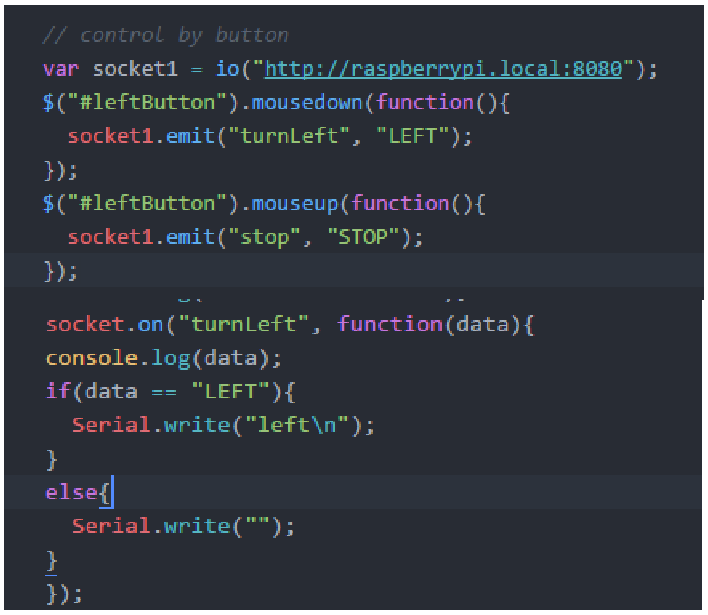

Introduction
Component
Control
Map
Components
List of components which the robot car uses. Click on the picture to see the datasheet of ea
3D camera sensor
Controller
Motor controller
Motor driver
Motor
Rear wheel
reset
Control
Click the buttons to control the direction of the car

Figure 34. Component tab and control tab

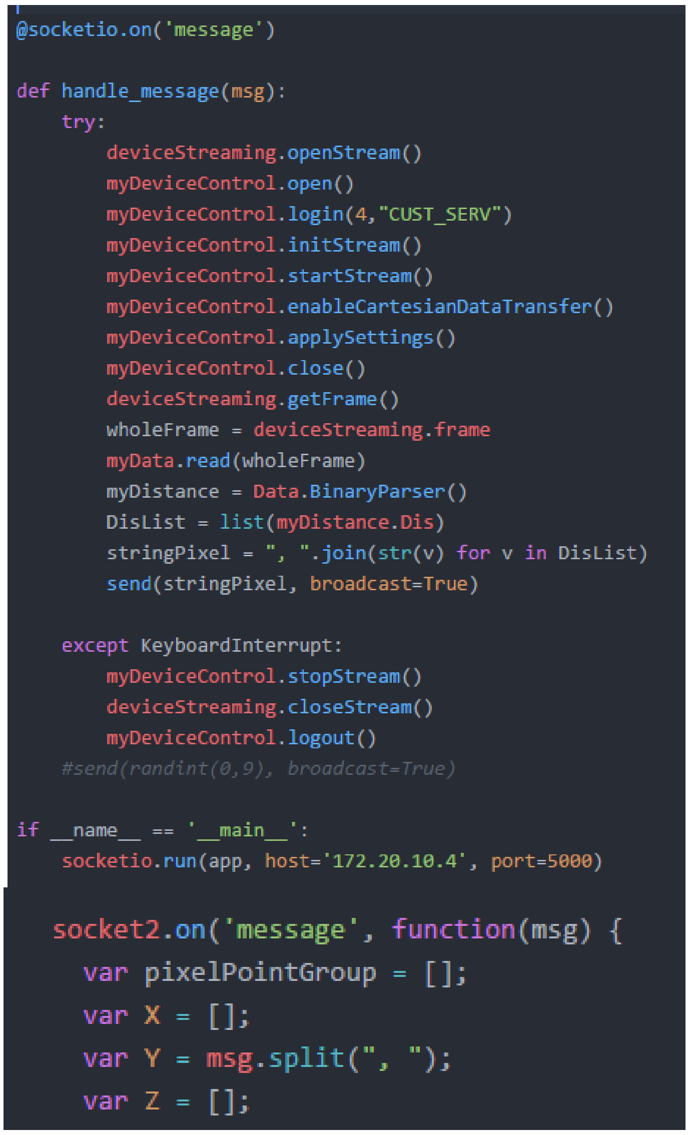
To send the controlling requests from the user to the car, it is necessary to use websockets to implement. WebSockets is a bidirectional communication technology for web applications. Its function is to operate over a single socket and to be exposed via a JavaScript interface in HTML 5 compliant browsers. If Websockets is connected with the web server, users can send data from browser to server by using a "send()" method, and receive data from server to browser by an "onmessage" event handler. In this platform, I used websockets to send the data from the 3D camera sensor and adjust the direction of the car. Figure 35 and Figure 36 show how to configure a web socket to send and receive data.



```
// control by button
var socket1 = io("http://raspberrypi.local:8080");
$("#leftButton").mousedown(function(){
  socket1.emit("turnLeft", "LEFT");
});
$("#leftButton").mouseup(function(){
  socket1.emit("stop", "STOP");
});
```

```
socket.on("turnLeft", function(data){
console.log(data);
if(data == "LEFT"){
  Serial.write("left\n");
}
else{
  Serial.write("");
}
});
```

Figure 35. HTML and Node.js server send and receive direction request



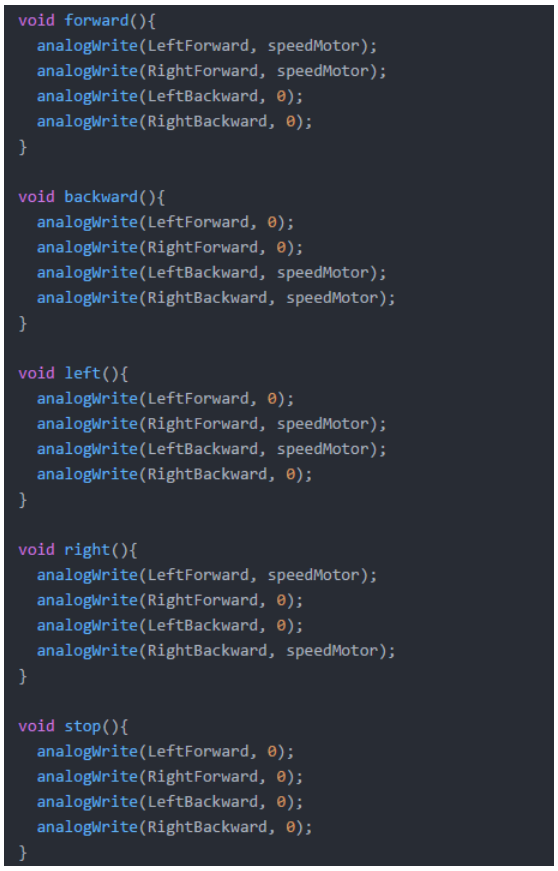
```
@socketio.on('message')

def handle_message(msg):
    try:
        deviceStreaming.openStream()
        myDeviceControl.open()
        myDeviceControl.login(4,"CUST_SERV")
        myDeviceControl.initStream()
        myDeviceControl.startStream()
        myDeviceControl.enableCartesianDataTransfer()
        myDeviceControl.applySettings()
        myDeviceControl.close()
        deviceStreaming.getFrame()
        wholeFrame = deviceStreaming.frame
        myData.read(wholeFrame)
        myDistance = Data.BinaryParser()
        DisList = list(myDistance.Dis)
        stringPixel = ", ".join(str(v) for v in DisList)
        send(stringPixel, broadcast=True)

    except KeyboardInterrupt:
        myDeviceControl.stopStream()
        deviceStreaming.closeStream()
        myDeviceControl.logout()
    #send(randint(0,9), broadcast=True)

if __name__ == '__main__':
    socketio.run(app, host='172.20.10.4', port=5000)
```

```
socket2.on('message', function(msg) {
  var pixelPointGroup = [];
  var X = [];
  var Y = msg.split(", ");
  var Z = [];
```

Figure 36. Python and HTML send and receive sensor data

3.4.3 Controlling motor

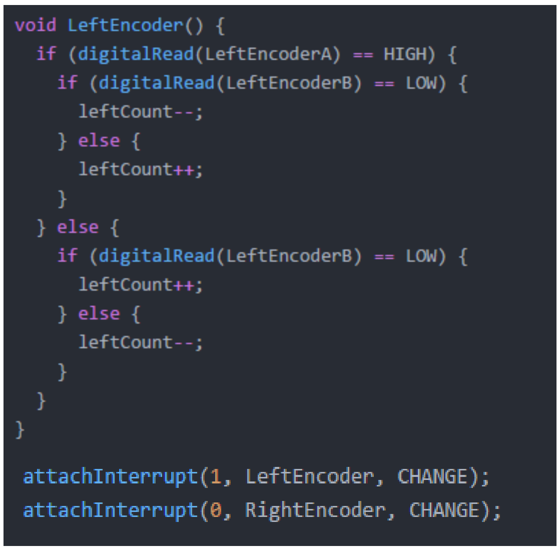
According to "Arduino" part in methodology, the main function of the microcontroller board is to control the speed of motor and read the encoder. Figure 37 shows the code of controlling direction of motors.



```
void forward(){
  analogWrite(LeftForward, speedMotor);
  analogWrite(RightForward, speedMotor);
  analogWrite(LeftBackward, 0);
  analogWrite(RightBackward, 0);
}

void backward(){
  analogWrite(LeftForward, 0);
  analogWrite(RightForward, 0);
  analogWrite(LeftBackward, speedMotor);
  analogWrite(RightBackward, speedMotor);
}

void left(){
  analogWrite(LeftForward, 0);
  analogWrite(RightForward, speedMotor);
  analogWrite(LeftBackward, speedMotor);
  analogWrite(RightBackward, 0);
}

void right(){
  analogWrite(LeftForward, speedMotor);
  analogWrite(RightForward, 0);
  analogWrite(LeftBackward, 0);
  analogWrite(RightBackward, speedMotor);
}

void stop(){
  analogWrite(LeftForward, 0);
  analogWrite(RightForward, 0);
  analogWrite(LeftBackward, 0);
  analogWrite(RightBackward, 0);
}
```

Figure 37. Controlling motor in Arduino

In the program of Arduino, encoder value is decoded by those command which is shown in Figure 38.



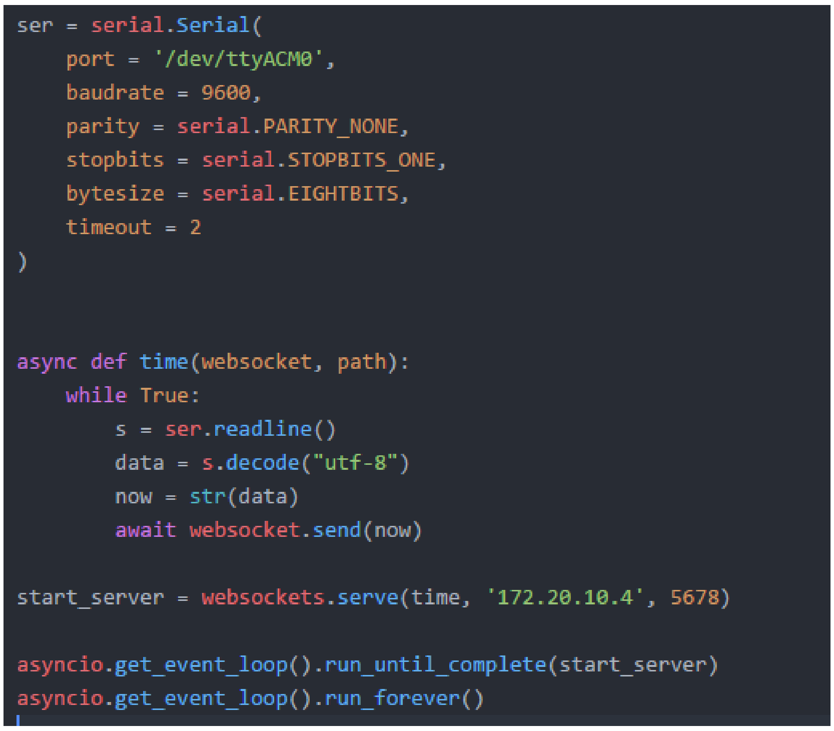
```
void LeftEncoder() {
  if (digitalRead(LeftEncoderA) == HIGH) {
    if (digitalRead(LeftEncoderB) == LOW) {
      leftCount--;
    } else {
      leftCount++;
    }
  } else {
    if (digitalRead(LeftEncoderB) == LOW) {
      leftCount++;
    } else {
      leftCount--;
    }
  }
}

 attachInterrupt(1, LeftEncoder, CHANGE);
 attachInterrupt(0, RightEncoder, CHANGE);
```

Figure 38. Decode the encoder value

According to the code in Figure 8, the Arduino counts the ticks of each channel A and B of the encoder by using the interrupt pins. The total of number of time which pulse of channel A and B are in HIGH and LOW state is indicated by variable “leftCount” or “rightCount”. As the datasheet of the motor, the count of each resolution of encoder is 64 CPR and the gear ratio is 30:1. As the result, the angle of the motor is calculated as function (4):

$$\text{Angle} = \frac{64*30}{2} * \frac{1}{360}\ (deg) \qquad (4)$$

After collecting the encoder data, Arduino sends the values to server by following commands which are shown in Figure 39.

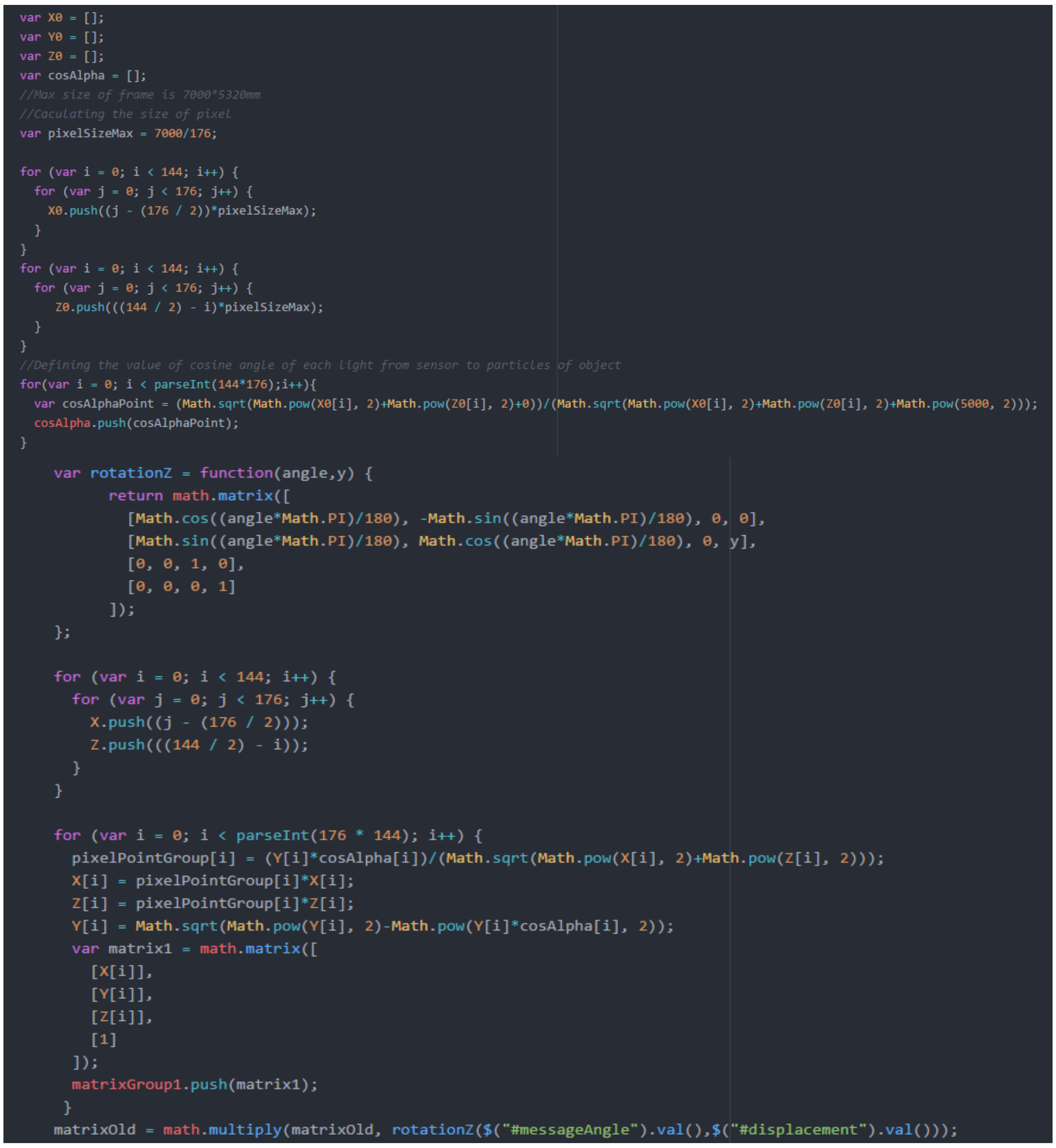


```
ser = serial.Serial(
    port = '/dev/ttyACM0',
    baudrate = 9600,
    parity = serial.PARITY_NONE,
    stopbits = serial.STOPBITS_ONE,
    bytesize = serial.EIGHTBITS,
    timeout = 2
)


async def time(websocket, path):
    while True:
        s = ser.readline()
        data = s.decode("utf-8")
        now = str(data)
        await websocket.send(now)

start_server = websockets.serve(time, '172.20.10.4', 5678)

asyncio.get_event_loop().run_until_complete(start_server)
asyncio.get_event_loop().run_forever()
```

Figure 39. Sending encoder data to server

3.4.4 Visualizing data

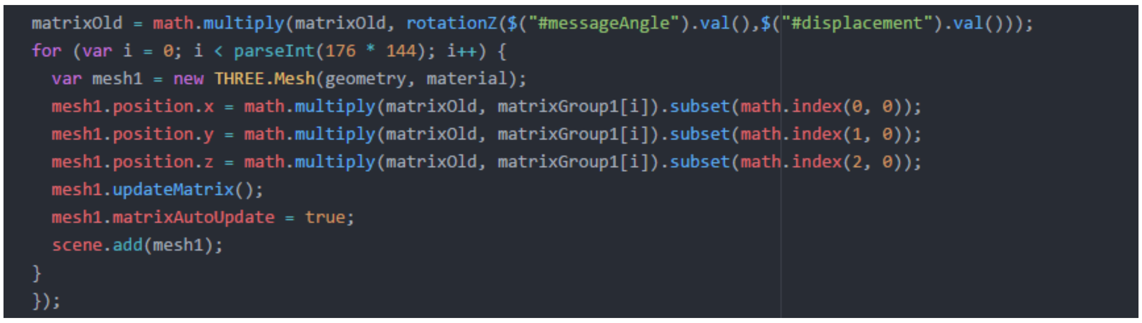
To visualize the position of particles which 3D camera sensor collected, there are three main steps: calculating the parameter X, Y and Z of each particle, navigating the position of the car and rendering the data. In the first step, according to the principle of 3D camera sensor, the distance from the sensor plane to the particle is calculated by function (3). Figure 40 shows the code applied the function.



```
var X0 = [];
var Y0 = [];
var Z0 = [];
var cosAlpha = [];
//Max size of frame is 7000*5320mm
//Caculating the size of pixel
var pixelSizeMax = 7000/176;

for (var i = 0; i < 144; i++) {
  for (var j = 0; j < 176; j++) {
    X0.push((j - (176 / 2))*pixelSizeMax);
  }
}
for (var i = 0; i < 144; i++) {
  for (var j = 0; j < 176; j++) {
     Z0.push(((144 / 2) - i)*pixelSizeMax);
  }
}
//Defining the value of cosine angle of each light from sensor to particles of object
for(var i = 0; i < parseInt(144*176);i++){
  var cosAlphaPoint = (Math.sqrt(Math.pow(X0[i], 2)+Math.pow(Z0[i], 2)+0))/(Math.sqrt(Math.pow(X0[i], 2)+Math.pow(Z0[i], 2)+Math.pow(5000, 2)));
  cosAlpha.push(cosAlphaPoint);
}
    var rotationZ = function(angle,y) {
          return math.matrix([
            [Math.cos((angle*Math.PI)/180), -Math.sin((angle*Math.PI)/180), 0, 0],
            [Math.sin((angle*Math.PI)/180), Math.cos((angle*Math.PI)/180), 0, y],
            [0, 0, 1, 0],
            [0, 0, 0, 1]
          ]);
    };

    for (var i = 0; i < 144; i++) {
      for (var j = 0; j < 176; j++) {
        X.push((j - (176 / 2)));
        Z.push(((144 / 2) - i));
      }
    }

    for (var i = 0; i < parseInt(176 * 144); i++) {
      pixelPointGroup[i] = (Y[i]*cosAlpha[i])/(Math.sqrt(Math.pow(X[i], 2)+Math.pow(Z[i], 2)));
      X[i] = pixelPointGroup[i]*X[i];
      Z[i] = pixelPointGroup[i]*Z[i];
      Y[i] = Math.sqrt(Math.pow(Y[i], 2)-Math.pow(Y[i]*cosAlpha[i], 2));
      var matrix1 = math.matrix([
        [X[i]],
        [Y[i]],
        [Z[i]],
        [1]
      ]);
      matrixGroup1.push(matrix1);
     }
    matrixOld = math.multiply(matrixOld, rotationZ($("#messageAngle").val(),$("#displacement").val()));
```

Figure 40. Calculating the position of particle

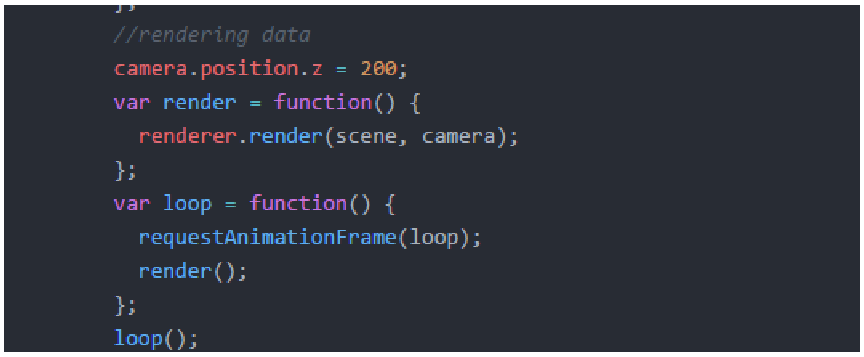
In the second step, because the car moves around the working environment, the position of particles will be changed as the displacement and rotation of the car. As the result, the matrix of particles' position must be multiplied with the matrix of rotation and displacement followed by equations (1), (2) and (4). Figure 41 shows codes calculating final position of the particles.



```
matrixOld = math.multiply(matrixOld, rotationZ($("#messageAngle").val(),$("#displacement").val()));
for (var i = 0; i < parseInt(176 * 144); i++) {
  var mesh1 = new THREE.Mesh(geometry, material);
  mesh1.position.x = math.multiply(matrixOld, matrixGroup1[i]).subset(math.index(0, 0));
  mesh1.position.y = math.multiply(matrixOld, matrixGroup1[i]).subset(math.index(1, 0));
  mesh1.position.z = math.multiply(matrixOld, matrixGroup1[i]).subset(math.index(2, 0));
  mesh1.updateMatrix();
  mesh1.matrixAutoUpdate = true;
  scene.add(mesh1);
}
});
```

Figure 41. Calculating the position of particle with displacement and rotation of the car

Finally, data is rendered by following commands in Figure 42



```
//rendering data
camera.position.z = 200;
var render = function() {
  renderer.render(scene, camera);
};
var loop = function() {
  requestAnimationFrame(loop);
  render();
};
loop();
```

Figure 42. Rendering data

4 RESULTS

4.1 Single 3D snapshot

In the map tab, the data of the particles' position is visualized. Figure 43 and Figure 44 show two images of one position. The first image was taken by a mobile phone and the second image was taken by the 3D camera sensor.

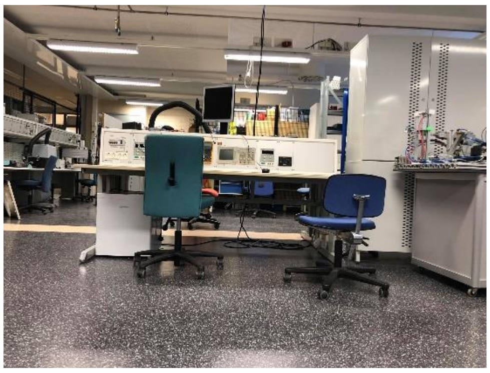


Figure 43. Scene taken by mobile phone camera

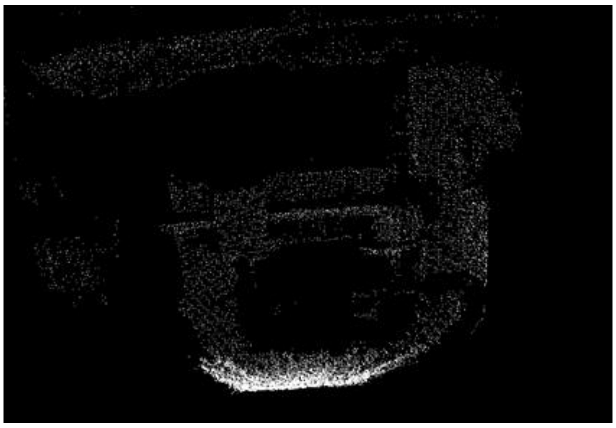


Figure 44. Scene taken by the camera

In the Figures, it can be seen that the second image shows a good resolution and the density of points is thick enough to recognize the objects in the scene (a table, chairs). However, with a wide range of angle, there are some particles missing. The limitation of the sensor is around 120 degrees at a maximum in a horizontal plane.

4.2 Multiple 3D snapshots

In this task, the car turned around 360 degrees and each time when it stopped, it captured the image of the opposite object. The result is shown in Figure 45.

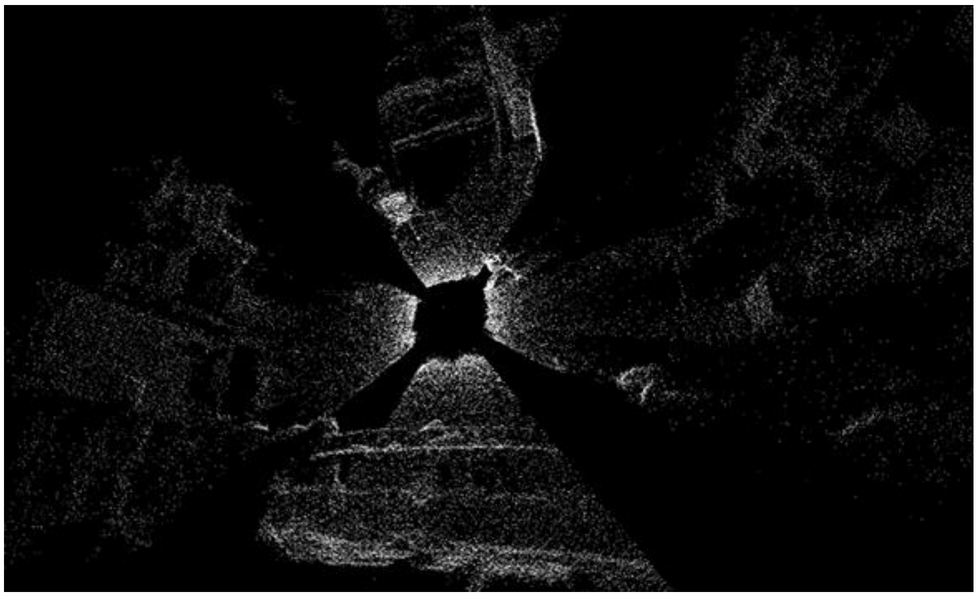


Figure 45. Scene in 360 degrees

The image shows the scene around the car and it can be seen that there are some empty spaces which the sensor could not reach. The disadvantage when controlling the car is that the encoder could not work accurately. As a result, the positions of some particles in the image were not correct. Moreover, because the number of points increased, the map was loaded slower and it was difficult to change the view of the map.

5 CONCLUSION

The robot car performed good movement when controlling its direction. The images of 3D snapshots were clear enough for viewing with small spaces. The advantage of this platform is that it can be used in vehicle industry where the developer can use pixel data from the sensor to program and manage groups of cars. In addition, this visualizing software can be applied to a working environment where there is a need to test and detect the geography as well as navigate.

However, there are some disadvantages with the program:

- The time of code execution is slow because of using high level languages such as JavaScript or Python.
- The library of rendering the data should be changed to optimize the time of rendering because of the limitation of the Raspberry Pi memory.

REFERENCES

Arduino-mega-2560-rev3. (n.d.). Retrieved from arduino: https://store.arduino.cc/usa/arduino-mega-2560-rev3

asd. (n.d.).

B00HV43UOG. (n.d.). Retrieved from amazon: https://www.amazon.de/gp/product/B00HV43UOG/ref=oh_aui_detailpage_o00_s00?ie=UTF8&psc=1

B077M59T75. (n.d.). Retrieved from amazon: www.amazon.de/gp/product/B077M59T75/ref=ox_sc_act_title_1?smid=A1HPO255OC5X3C&psc=1

Barr, M., & Massa , A. (1999). *Programming embedded systems: with C and GNU development tools.* O'Reilly.

Barrett, S. (2009). *Embedded Systems Design with Atmel AVR Microcontroller Part 1.* Laramie: Morgan & Claypool.

Blurb. (n.d.). Retrieved from Python: https://www.python.org/doc/essays/blurb/

CID Visionary-T AG. SICK. (n.d.).

Communication interface description. SICK. (n.d.).

CSS. (n.d.). Retrieved from W3Schools: www.w3schools.com/css/tryit.asp?filename=trycss_syntax_element

Exide Litium akku 12V ELTZ14S. (n.d.). Retrieved from motonet: http://www.motonet.fi/fi/tuote/900443/Exide-Litium-akku-12V-ELTZ14S

Flanagan, D. (1998). *JavaScript: The definitive guide.* Sebastoppol: O'Reilly.

Greicius, T. (n.d.). *overview*. Retrieved from nasa: https://www.nasa.gov/mission_pages/msl/overview/index.html

How 3D Imaging works. (n.d.). Retrieved from visiononline: www.visiononline.org/blog-article.cfm/How-3D-Imaging-Works/44

Intelligence, S. S. (n.d.). *www.youtube.com*. Retrieved from www.youtube.com/watch?v=_HoQwItJ3ks

Introduction. (n.d.). Retrieved from arduino: www.arduino.cc/en/Guide/Introduction

Jargon. (2017, December 20). Retrieved from computerhope: https://www.computerhope.com/jargon/h/html.htm

Javascript introduction. (n.d.). Retrieved from Mozilla: www.developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction

Jonathan, J. (2000). *Embedded Microcomputer Systems: Real Time Interfacing.* Stamford: CENGAGE Learning.

Lima, P., & Ribeiro, M. I. (2002, March). *Instituto superior technico.* Retrieved from Mobile robotics introduction: http://users.isr.ist.utl.pt/~mir/cadeiras/robmovel/Introduction.pdf

McManus, S., & Cook, M. (2013). *Raspberry Pi for dummies.* New Jersey: John Wiley & Sons, Inc.

microcontroller. (n.d.). Retrieved from techopedia: https://www.techopedia.com/definition/3641/microcontroller

Microprocessor overview. (n.d.). Retrieved from tutorialspoint: https://www.tutorialspoint.com/microprocessor/microprocessor_overview.htm *mobile robotics*. (n.d.). Retrieved from techopedia: www.techopedia.com/definition/33065/mobile-robotics

Motor driver. (n.d.). Retrieved from futureelectronics: www.futureelectronics.com/en/drivers/motor-driver.aspx

Niederst, J. (2001). *Web design in a nutshell.* California: O'Reilly.

product. (n.d.). Retrieved from cqrobotshop: www.cqrobotshop.com/index.php?route=product/product&product_id=1057&search=DC+motor+encoder&description=true

Product briefs. (n.d.). Retrieved from raspberrypi: https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf

quadrature encoder. (n.d.). Retrieved from dynapar: www.dynapar.com/technology/encoder_basics/quadrature_encoder/

Rawashdeh, M. (2014, April 29). *Motor driver BTS7960 43A*. Retrieved from instructables: http://www.instructables.com/id/Motor-Driver-BTS7960-43A/

Santiago, E. (2017, July 24). *embedded system*. Retrieved from mazingstudiotechnology: www.amazingstudiotechnology.com/embedded-system/

Shannon, R. (2012, August 21). *What is HTML*. Retrieved from yourhtmlsource: www.yourhtmlsource.com/starthere/whatishtml.html

Shrestha, S. (2017, 3 2). *www.computersciencementor.com*. Retrieved from www.computersciencementor.com/advantages-and-disadvantages-of-wi-fi/

Visionary T. (n.d.). Retrieved from SICK: https://www.sick.com/fi/en/vision/3d-vision/visionary-t/c/g358152

Volosyak, I. (2016). *Microcontroller lecture 1.* Kleve.

web design. (n.d.). Retrieved from w3: www.w3.org/standards/webdesign/htmlcss#whatcss

What is a Raspberry Pi. (n.d.). Retrieved from raspberrypi: www.raspberrypi.org/help/what-%20is-a-raspberry-pi/