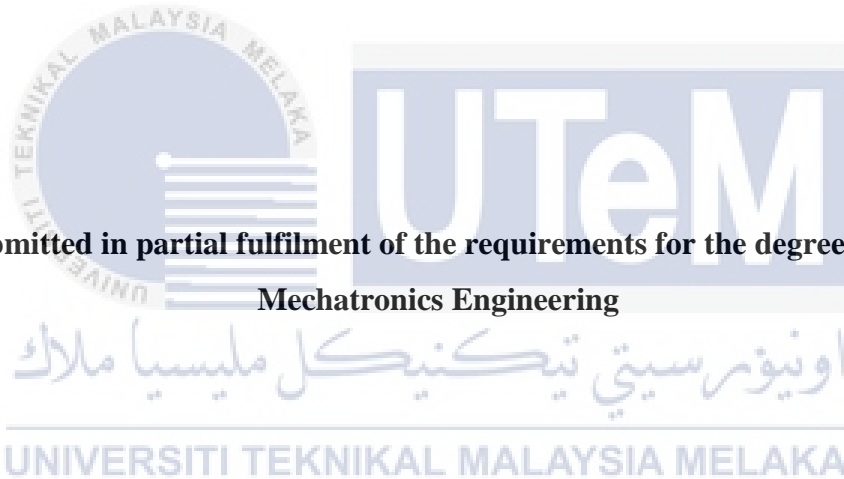


**AUTOMATED WASTE SEPARATED MACHINE**

**LOW XIAN MIN**



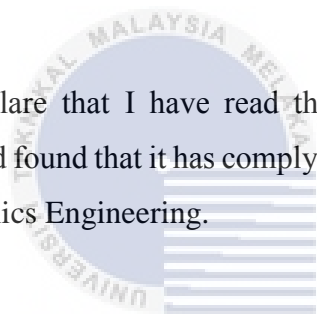
**Faculty of Electrical Engineering**

**UNIVERSITI TEKNIKAL MALAYSIA MELAKA**

**2017**

### SUPERVISOR ENDORSEMENT

I hereby declare that I have read through this report entitle “Automated Waste Separated Machine” and found that it has comply the partial fulfilment for awarding the degree of Bachelor of Mechatronics Engineering.



Signature  : .....

Supervisor's Name : .....

Date : .....

## DECLARATION

I declare that this report entitled “Automated Waste Separated Machine” is the result of my own research except as cited in the references. The report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature : .....  
Name : .....  
Date : .....

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## DEDICATION

To my beloved mother and father



## ACKNOWLEDGEMENT

First of all, I am grateful to Guatama Buddha for giving me the strength and health all the time, especially during the process of doing my Final Year Project in my final semester.

The way to do this report, I was in contact with many seniors from UTeM, researchers, academicians and practitioners. They had contributed towards my understanding and thought. I would like to take this chance to express my sincere appreciation to my main project supervisor, Pn. Nursabillilah Binti Mohd Ali, for encouragement, guidance critics and friendship. Without her continued support and interest, this project would not have been same as presented here.

I would also like to thank my girl-friend and housemates and others who have helped at various occasions. Their views and tips are useful indeed. Unfortunately, it is not possible to list all of them in this limited space. Last but not least, I am grateful to all my family members who trust and support me all the time.

Last but not least, I would like to thank my family members, Low Chee Beng (Father), Wong Choi Chun (Mother), Low Pei Pei and Low Pei Yee (Sisters) for supporting me 24 years in mentality or physically. Without them, I cannot perform so well to reach this stage. They always trust me and give me the strength to move forward without any fear.

## ABSTRACT

Beverage container waste has been brought serious impacts to our environmental. For example, energy consumption of manufacturing drinking containers equivalences to 30 to 50 million barrels of crude oil each year. Besides that, process of making these containers generate large amount of greenhouse gas that causes global warming. Moreover, toxic is emitted to the air and water from the process of turn bauxite into alumina. Undeniable, demand of using drinking container is keep on rising, which cannot be eliminated. Hence, recycle is a method to be used to lower these impacts to the environment by controlling the quantity of drinking container available in our surrounding. Common way to perform recycle activity is using manually picking and sorting drinking bottle into categories but it is hard to solve the problem in long term due to high rate of beverage production and high cost to manage waste generated every day. Thus, machine vision is proposed to solve the problem.

Machine vision is one of the popular technique for object detection. The extent of this project is using machine vision to recognise type of waste (Note: Usual shape of drinking bottle) by extracting the colour of waste. Colour value (RGB) is transformed into HSV (Hue, Saturation and Value). Histograms are created for comparison between detected object and saved object in database. Then, sort them into different place by using machine learning. OpenCV library play a crucial role in this project, which is included in Microsoft Visual Studio 2012 with C++ language to write the coding for waste recognition.

As a summary, the machine vision colour detection is used to build Automated Waste Separated Machine and able to achieve 80% rate of success of waste detection.

## ABSTRAK

Sisa bekas minuman telah membawa kesan yang serius kepada alam sekitar. Sebagai contoh, penggunaan tenaga bekas minum pembuatan persamaan yang 30 hingga 50 juta tong minyak mentah setiap tahun. Selain itu, proses membuat bekas ini menjana jumlah besar gas rumah hijau yang menyebabkan pemanasan global. Terlebih dahulu, toksik dipancarkan ke udara dan air daripada proses seterusnya bauksit menjadi alumina. Tidak dapat dinafikan, permintaan menggunakan bekas minum adalah terus meningkat, yang tidak boleh dihapuskan. Oleh itu, kitar semula adalah satu kaedah yang akan digunakan untuk mengurangkan kesan ini kepada alam sekitar dengan mengawal kuantiti bekas minuman terdapat di sekeliling kita. Cara yang sama untuk melakukan aktiviti kitar semula menggunakan secara manual memilih dan menyusun botol minuman ke dalam kategori tetapi ia adalah sukar untuk menyelesaikan masalah dalam jangka panjang kerana kadar yang tinggi pengeluaran minuman dan kos yang tinggi untuk menguruskan sisa yang dihasilkan setiap hari. Oleh itu, penglihatan mesin dicadangkan untuk menyelesaikan masalah tersebut.

Penglihatan mesin adalah salah satu teknik yang popular bagi pengesanan objek. Sejauh mana projek ini menggunakan penglihatan mesin untuk mengenali jenis bahan buangan (Nota: bentuk Usual botol minum) dengan mengeluarkan warna sisa. nilai warna (RGB) berubah menjadi HSV (Hue, Saturation dan Undian). Histogram yang dicipta untuk perbandingan antara objek yang dikesan, dan objek yang disimpan dalam pangkalan data. Kemudian, menyusun mereka ke dalam tempat yang berbeza dengan menggunakan pembelajaran mesin. perpustakaan OpenCV memainkan peranan penting dalam projek ini, yang termasuk dalam Microsoft Visual Studio 2012 dengan C ++ bahasa untuk menulis kod untuk pengiktirafan sisa.

Sebagai ringkasan, pengesanan warna penglihatan mesin digunakan untuk membina Automated Waste Dipisahkan Mesin dan mampu mencapai kadar 80% kejayaan pengesanan sisa.

## TABLE OF CONTENTS

<b>CHAPTER</b>	<b>TITLE</b>	<b>PAGE</b>
	<b>SUPERVISOR ENDORSEMENT</b>	<b>i</b>
	<b>DECLARATION</b>	<b>iii</b>
	<b>DEDICATION</b>	<b>iv</b>
	<b>ACKNOWLEDGEMENT</b>	<b>v</b>
	<b>ABSTRACT</b>	<b>vi</b>
	<b>ABSTRAK</b>	<b>vii</b>
	<b>TABLE OF CONTENTS</b>	<b>viii</b>
	<b>LIST OF TABLES</b>	<b>xi</b>
	<b>LIST OF FIGURES</b>	<b>xii</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 Motivation	1
	1.2 Problem Statement	2
	1.3 Objective	3
	1.4 Scope	3
<b>2</b>	<b>LITERATURE REVIEW</b>	<b>4</b>
	2.1 Introduction	4
	2.2 Mechanical technique of waste sorting	7
	2.2.1 Manually recycle technique	7
	2.2.2 Infrared camera technique	8
	2.2.3 Near infrared and charged couple device camera technique	9
	2.2.4 Inductive Sensor Array and Colour Vision	10
	2.2.5 Pick and place technique	11



2.3	Image processing technique	14
2.3.1	Introduction	14
2.3.2	Histogram in image processing	15
2.3.3	Logic Operation and Morphological in image processing	19
2.3.4	Segmentation in image processing	21
2.3.5	Colour in image processing	23
2.3.6	Decision Based on Neighbourhood White Strip	27
2.3.7	Object Shape Recognition	29
2.4	Summary of chapter	32
<b>3</b>	<b>METHODOLOGY</b>	<b>33</b>
3.1	Determine possible shape, perimeter and colour of bottles.	33
3.2	Hardware development	34
3.3	Software Overview and Selection	35
3.3.1	VXL	35
3.3.2	LTI	35
3.3.3	OpenCV	36
3.4	Write coding for comparison and actuation between capture image and database image	39
3.5	Proof of concept by using Visual Studio C++ and Arduino	39
3.6	Electric circuit development	40
3.7	Development of full structure of mechanism	40
3.8	Proof of the Machine Vision with full structure of actuator	41
3.9	Experiment	42

3.1	Flow chart of Automated Waste Separated Machine	44
3.11	Flow of activities	45
<b>4</b>	<b>RESULT AND DISCUSSION</b>	<b>46</b>
4.1	Introduction of image processing result	46
4.1.1	Result from glass bottle test	51
4.1.2	Discussion from glass bottle test	56
4.1.3	Result from plastic bottle test	58
4.1.4	Discussion from plastic bottle test	63
4.1.5	Result from aluminium can test	65
4.1.6	Discussion from aluminium can test	70
4.1.7	Result from overall accuracy testing	72
4.1.8	Discussion from final test (mixed bottle)	76
<b>5</b>	<b>CONCLUSION AND RECOMMANDATION</b>	<b>77</b>
5.1	Conclusion	77
5.2	Recommendation	79
<b>6</b>	<b>REFERENCES</b>	<b>80</b>
<b>7</b>	<b>APPENDIX</b>	<b>83</b>

## LIST OF TABLES

TABLE	TITLE	PAGE
2.1	Comparison between Traditional and Recycle Waste Treatment	5
2.2	Mechanical technique overview	13
2.3	Image processing technique overview	30
3.1	Comparison between different types of machine vision software	37
4.1	Glass test 1 of ten input	51
4.2	Glass test 2 of ten input	52
4.3	Glass test 3 of ten input	53
4.4	Plastic (DESA) test 1 of ten input	58
4.5	Plastic (100 PLUS) test 2 of ten input	59
4.6	Plastic (DESA & 100 PLUS) test 3 of ten input	60
4.7	Can (Pepsi) test 1 of ten input	65
4.8	Can (Milo) test 2 of ten input	66
4.9	Can (Milo & Pepsi) test 3 of ten input	67
4.10	Final (mixed bottles) test 1 of ten input	72
4.11	Final (mixed bottles) test 2 of ten input	73
4.12	Final (mixed bottles) test 3 of ten input	74

## LIST OF FIGURES

TABLE	TITLE	PAGE
2.1	Labourers differentiate the waste manually [9]	7
2.2	Spectra for HDPE, PP, PET and PS by using NIR [11]	8
2.3	Colour feature extraction for a plastic bottle in upright position image by machine vision [13]	9
2.4	Two-dimensional classification [12]	10
2.5	Close loop system [14]	11
2.6	a) Pick and place robot, b) Gripper [13]	12
2.7	Nine key stages in digital image processing [15]	15
2.8	Image enhancement by using histogram equalisation [15]	16
2.9	All five segmented regions and generated ROIs [16]	17
2.10	Bounding box image [16]	18
2.11	Block diagram of plastic bottle classification [16]	18
2.12	Image enhancement using AND operator [17]	19
2.13	Image enhancement using OR operator [17]	19
2.14	Set A, Structuring Element B, A eroded by B and Boundary machines of boundary extraction [17]	20
2.15	Result of using boundary extraction algorithm from binary image [17]	20
2.16	3x3 mask [15]	21
2.17	Initial population of 50 lines [18]	22
2.18	Final population and the thick black line (best-fitting element) [18]	22
2.19	Hue and saturation in the HSI colour model [17]	24
2.20	Concept of colour classification by using fuzzy and genetic algorithm [19]	26
2.21	The chromaticity graph of different type of glass bottles [19]	26
2.22	Flow of plastic sorting [20]	27

	(a) Input image, (b) Extracted foreground, (c) White strips, (d) Contour	
2.23	boxes [20]	28
2.24	8-connected box for each white strip [20]	28
2.25	Example of shape detection [21]	29
3.1	Side view	40
3.2	Top view	40
3.3	Front view	40
3.4	Functionality testing of mechanism with LED light up	41
3.5	Console showed value of comparison of aluminium can	42
3.6	System flow chart	44
4.1	Method to compare images from camera and databased	50
4.2	Histogram of glass test result of 30 input	54
4.3	Glass was ready to be transferred	55
4.4	Glass rotated clockwise 225 degree	55
4.5	Histogram of 30 input plastic test result	61
4.6	Plastic bottle was ready to be transferred	62
4.7	Plastic bottle rotated clockwise 135 degree	62
4.8	Histogram of 30 input can test result	68
4.9	Aluminium tin was ready to be transferred	69
4.10	Aluminium tin rotated clockwise 180 degree	69
4.11	Histogram of 30 input mixed bottles test result	75

## CHAPTER 1

### INTRODUCTION

#### 1.1 Motivation

Half a million years ago, human created very less garbage, the main reason was that human wasted very little. They reused and repaired most of what they had rather than replaced. When came to the Revolution of Industrial, which began in early 1700s, large amount of waste was produced. Products were made by machine rather than hand make, which meant that more goods could be made in cheaper way and could be produced in large amount [1].

Nowadays, they are a lot of beverage available in market like Coca-Cola and Pepsi. The common types of container to store the drink are plastic bottles, aluminium cans and glass bottles. People tend to throw away these empty bottles or cans after use. Without a good management to recycle all these bottles and cans, number of waste will be increased significantly. One day, natural resources will run out soon. A good waste management is a key point to build a sustainable community. Pollution can be reduced and resource can be preserved by recycling some key resource such as metal, glass and plastic that have been thrown in landfill. Therefore, recycle plays an important role to conserve our natural resources.

In China, around 3818 tons of metal, 11820 tons of glass, and 110190 tons of plastic are thrown away in 2006, these 3 type all waste cover around 14% of overall waste [2]. Furthermore,

Singapore also generates a huge volume of waste every day, which can cover up to 1030 football fields with a normal people height in a year [3].

In Malaysia, government enforced the law to sort all the waste in category, but there are a lot of users fail to do that. In statistic from Malay Mail Online, there are around 3000tonnes of waste being throw away in every single day in Malaysia [4].

In state of Melaka, statement shows that each resident generates 250kg of waste per year. The population of this state is around 872,900, it means that 218225tons of waste are thrown in a year [5]. All the statement shows that an effective way to separate waste should be invented to solve waste problem for better future. It is a main factor to build a sustainable community.

## 1.2 Problem Statement

Nowadays, there are a lot of soft drink and mineral water are served with plastic bottle, aluminium can and glass to make sure these drink can transport and sell in everywhere. Besides that, using these containers to store drink can ensure quality of taste and prevent metamorphic problem. There are a lot of favour drink sell in Malaysia like Pepsi, Milo, 100plus, Desa mineral and etc. Some of these drink fill in aluminium can, some fill in glass bottle and plastic bottle. All of it are designed and manufactured with different colour. If 6 people out of 10 in Malaysia buy at least one soft drink for their own daily, a huge number of waste are created everyday if proper waste management is not applied.

Furthermore, unbent drinking container waste normally occupy a lot of space in public dustbin. This is one of the reason of making dustbin overflow and smudge our surrounding. Manually picking and sorting for bottle into categories is not sufficient to encounter the problem in long term due to high rate of beverage production and high cost to manage waste generated. There are also no sensors can detect the property of bottle.

By solving this matter, initiative has to be taken to minimise the impacts bring to life. Using machine vision to replace human eye to detect and using Arduino Uno pair with gripper to sort all these bottles is one of a solution to perform the heavy task daily. For instance, plastic bottle and cans have its own unique colour and reflectivity of light. By using object's colour and

reflectivity of light comparison algorithm, machine vision technique able to perform object identification just like human. Furthermore, proper lightning and suitable position of camera are the key factors for less noise detection.

### 1.3 Objective

The objective of the research included:

1. To recognise object based on machine vision colour detection.
2. To design a mechanism to separate unbend waste bottle.
3. To analysis system performance.

### 1.4 Scope

Based on the process of designation and the consideration of limited time, the scopes of this project are listed as:

1. Focus on unbend or partially bent waste drinking bottles.
2. Emphasize on separation between 500ml 100 plus and DESA plastic bottles, 325ml brown colour glass bottles and 325ml aluminium cans.
3. The developed algorithms are used for unbent or partially bent waste bottle or can detection only.
4. A non-moveable camera is attached in a partially cover black box for bottle and can detection.
5. Open loop pick and place gripper can only pick, release and rotate 135, 180 and 225 degrees.
6. Performance of actual test will be affected by light intensity and reflective of light.



## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Introduction

In China, landfilling is a technique for municipal solid waste (MSW) management. There is roughly 80~90 percent of the waste produced are disposed by this method. Based on their performed literature survey and analysis on this title, they come out the conclusion of the best way is to control the pollution source and cut off the pollution pathway [6]. This method of sorting does not consider the category of waste. Waste is simply thrown away after use and place in a place called landfill, which is the simplest way to solve waste management problem, but bring out a massive problem to the environment. For example, if the landfill without any landfill cap, leachate production will be increased. Thus, methane  $\text{CH}_4$  emission cannot be decreased.

In the past decade, some developing countries pushing the speed of economic is the priority and ignored environmental problem is impossible to avoid. Hence, the whole world today pays more attention on wastes recycling as there is huge amount of wastes being thrown away day by day that cause a lot of problems like health issue and greenhouse effect. Recycle is a process which can convert wastes into useable materials. One of the outcomes from investigation shown that current waste treatment is not enough to recycle waste generated every day by just using man power. The result illustrated the highest rate of factor choosing waste disposal way is based on convenient which was rated as 3.36 [7]. Table 2.1 (Comparison between Traditional and Recycle Waste Treatment) below illustrated the summary between traditional waste treatment and recycled waste treatment.

Table 2.1: Comparison between Traditional and Recycle Waste Treatment

NO	Waste Treatment Criteria	Traditional Waste Treatment	Recycle Waste Treatment
1.	Way to throw waste	Mix all waste and simply throw in dustbin.	Separate waste in categories and throw in certain platform.
2.	Processing waste	<ol style="list-style-type: none"> <li>1. Landfilling</li> <li>2. Incineration</li> </ol>	<ol style="list-style-type: none"> <li>1. Reuse</li> <li>2. Reprocess to make new item</li> <li>3. As fertilizer</li> <li>4. Incineration</li> </ol>
3.	Waste category	No (All mixed)	<ol style="list-style-type: none"> <li>1. Recyclable</li> <li>2. Kitchen</li> <li>3. Harmful</li> <li>4. Other</li> </ol>
4.	Advantages	<ol style="list-style-type: none"> <li>1. Save cost in waste treatment.</li> <li>2. Save man power to separate waste.</li> <li>3. Less process for waste treatment.</li> <li>4.</li> </ol>	<ol style="list-style-type: none"> <li>1. Save natural resources.</li> <li>2. Decrease environment health issue.</li> <li>3. Produce sustainable community.</li> </ol>
5.	Disadvantages	<ol style="list-style-type: none"> <li>1. A lot of natural sources are wasted after used.</li> </ol>	<ol style="list-style-type: none"> <li>1. More and complicated steps are taken for waste treatment.</li> <li>2. Increase cost for waste treatment.</li> </ol>

		<ol style="list-style-type: none"><li>2. Methane CH<sub>4</sub> emission occurred at landfilled.</li><li>3. Greenhouse effect occurred due to methane gas.</li><li>4. Endanger health of wildlife.</li></ol>	<ol style="list-style-type: none"><li>3. More labourers are employed to sort waste manually.</li><li>4. Risk the life of labourers in manually sorting process.</li></ol>
--	--	--	---



اونيورسي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## 2.2 Mechanical technique of waste sorting

In the following sections, the reviews included different kind of mechanical technique of waste sorting. These sections of study may provide the general ideas for this Final Year Project (Automated Waste Separated Machine) moves smoothly.

### 2.2.1 Manually recycle technique

There are several sorting methods to extract plastic from waste. The most common method was employing labourers to differentiate the waste manually in order to get out plastic material as shown in Figure 2.1 below. This was an easy way to identify target material, but, it exposed labourers to hazardous Health Care Waste (HCW), which were potentially at risk of being infected and injured. According to World Health Organization, under section 1.2.1 Occupational and public health risks, title number 8, during handling of wastes, the medical and ancillary staff as well as the sanitary labourers could be injured if the waste had not been packed safely. For example, sharp components were considered as one of the most dangerous category of waste. Many injuries occur because syringe needles or other sharps had not been collected in safety boxes or because these had been overfilled. On dumpsites, scavengers during their recycling activities may also come in contact with infectious waste if it has not been properly treated or disposed [8].



Figure 2.1: Labourers differentiate the waste manually [9]

### 2.2.2 Infrared camera technique

Other than using manpower, one of the technique was using infrared camera [10]. The method to identify near infrared (NIR) spectra of plastic material was provided in Figure 2.2. From this spectrum, a coefficient set was obtained by using wavelet analysis. After that, coefficients were used to form a quaternion number. This number was going to be compared with standard value in order to determine the plastic material, which provide more detail about the plastic. The advantages of this technique included of robust and insensitivity to the noise of the signal. However, this method was less accurate due to it based on a simple Euclidean distance.

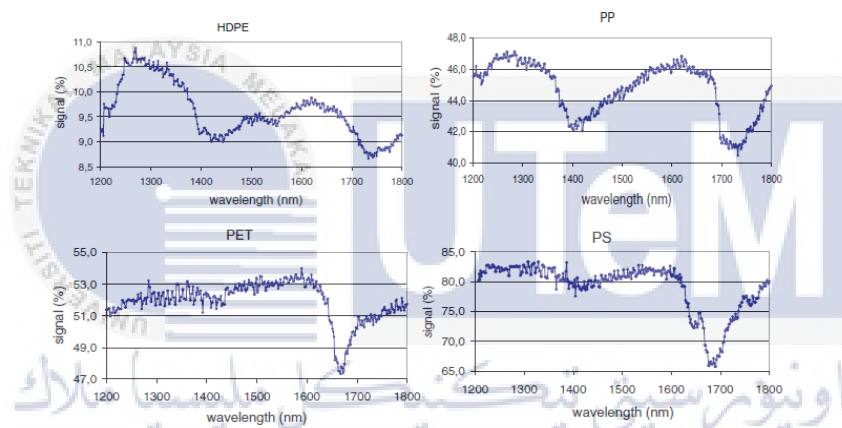


Figure 2.2: Spectra for HDPE, PP, PET and PS by using NIR [10]

### 2.2.3 Near infrared and charged couple device camera technique

Turning to the next point, a system that consisted two classification stages was developed [11]. The first stage is using Near Infrared (NIR) Spectroscopy to determine the type of plastic, which similar with the method of using infrared camera. The second stage was using machine vision based on a Charged Couple Device (CCD) camera. In addition, this method also integrated with quadratic discriminant function based classifier and decision tree classifier to increase overall colour classification accuracy as shown in Figure 2.3. This was probably due to it differed from one classifier to another for same colour. Unfortunately, this technique needed the bottle to be upright position, so that the sample could be scanned with minimal noise. Furthermore, this system become slower since the principle component analysis is added to do the adjustment.

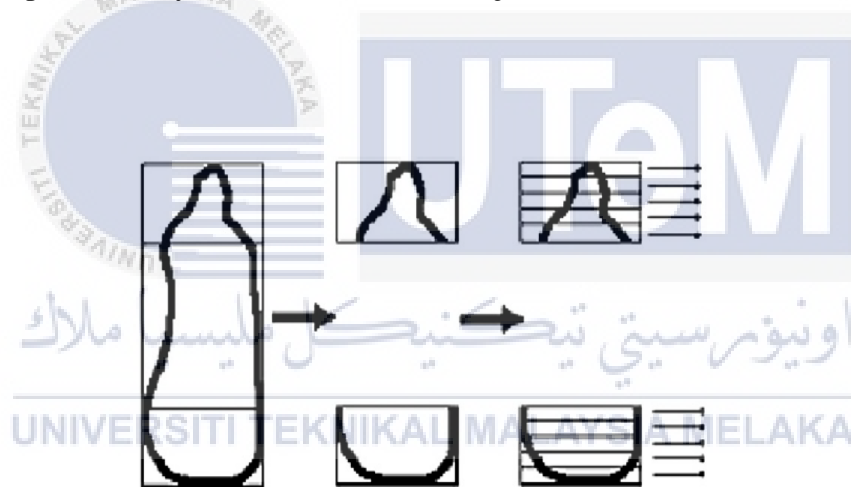


Figure 2.3: Colour feature extraction for a plastic bottle in upright position image by machine vision [11]

### 2.2.4 Inductive Sensor Array and Colour Vision

A sorting system for recycle metal scrap based on colour vision and inductive sensor was presented [12]. It combined two systems in a function to increase the ability of sensing different type of metal scraps. The colour detection was employing red channel as the common component which make comparison between green and blue value in order to find out the differences of colour as shown in Figure 2.4. Fluorescent bulb with high quality and efficiency was used to reduce the noise. Moreover, inductive system was used to measure the electrical properties of the metal for further confirmation of the metal properties. Hence, it abled to differentiate metals like steel, aluminium, copper and brass. However, the inductive system consisted of 52 sensors that performed the sensing function which not only increased the manufacturer cost but also raised the difficulty of machine learning. Sometimes, it was sufficient to use only machine vision to separate metals by just detect the colour them.

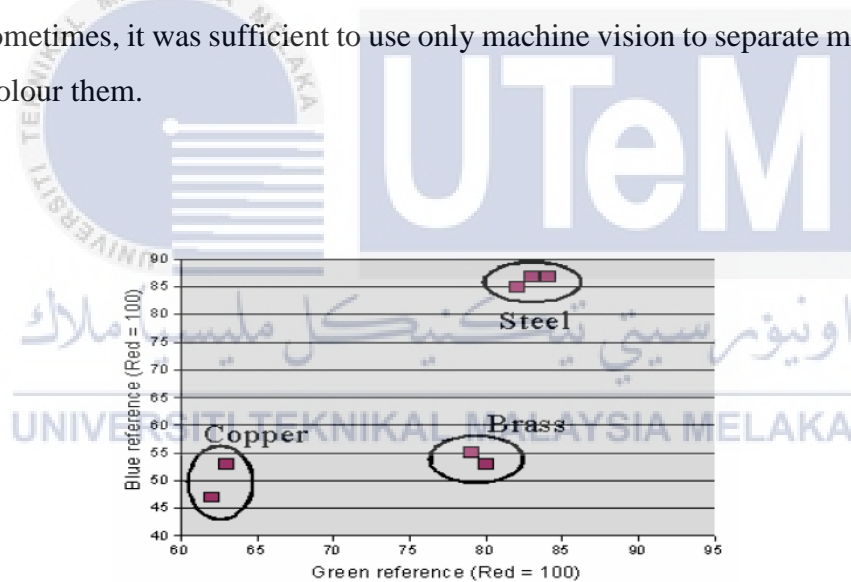


Figure 2.4: Two-dimensional classification [12]

### 2.2.5 Pick and place technique

Robot consisted three axes which let the robot to rotate in clockwise or counter clockwise, arm to move up or down and to extend and retract [13]. Some more, it consisted of gripper to that could open and close. The common actuators used was pneumatic cylinder that control by solenoid valve. Limit switch was included to trace the motion of cylinder. An example of operation shown the cylinder extended, its basement will rotate clockwise while retracted was another way around. The same cylinder not only could make rotational motion but also translational motion like arm extension. Easily speed control of the movement was one of the advantages of using pneumatic cylinder. An optional actuator included hydraulic cylinder. It used fluid compression instead of air compression. Comparison was made between pneumatic and hydraulic system. It illustrated that hydraulic system is more difficult to maintain due to its fluid leakage. Furthermore, it is more danger due to the leakage can easily cause fire [14].

Close-loop control was normally used in pick and place robot system. A feedback loop was included in the system to make adjustment of robot limb all the time. When there was difference between require position and limb position, the controller makes adjustment until the position of limb reaches to the desire position. The closed-loop arrangement was shown in Figure 2.5 below.

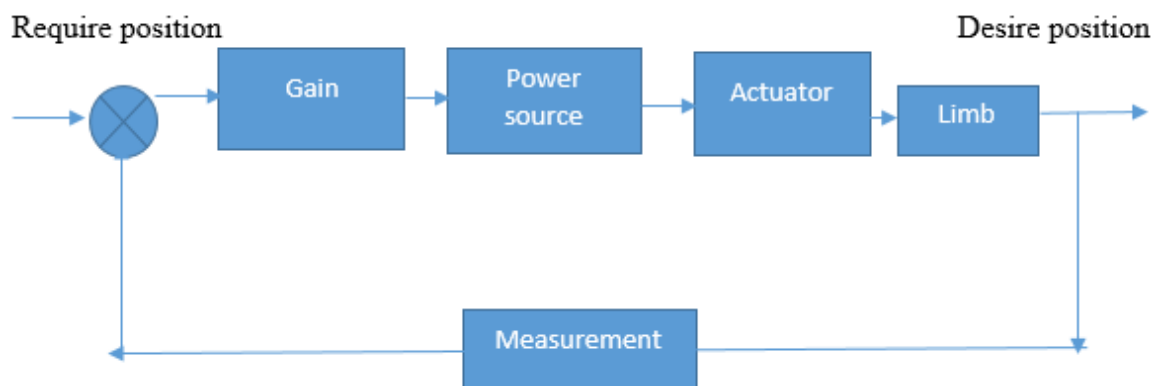


Figure 2.5: Close loop system [14]



The require force to move the end of limb can be calculated by using the equation:

$$F = G_c G_a (y_s - y) = m \frac{d^2 y}{dt^2} + k \frac{dy}{dt} \quad (1)$$

Where,  $y_s$  is set position

$y$  is actual position

$y_s - y$  is error signal between set position and actual position

$G_c$  &  $G_a$  are the gain of the controller

$m$  is total mass

$\frac{d^2 y}{dt^2}$  is acceleration

$k \frac{dy}{dt}$  is frictional force

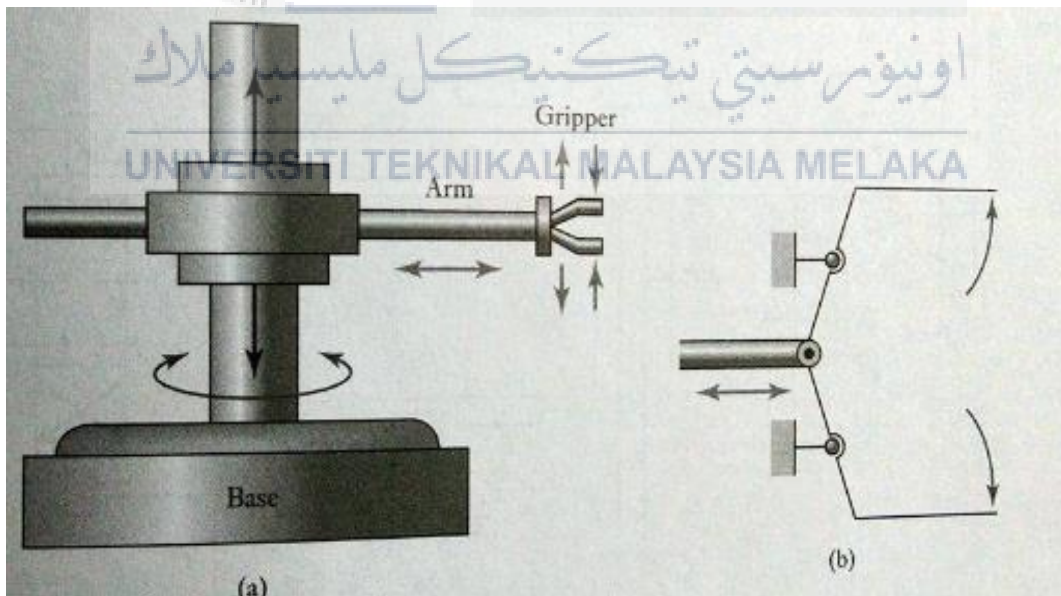


Figure 2.6: a) Pick and place robot, b) Gripper [13]

Table 2.2: Mechanical technique overview.

	Manually recycle technique [8]	Infrared camera technique [10]	Near infrared and charged couple device camera technique [11]	Inductive Sensor Array and Colour Vision [12]	Pick and place technique [13]
Advantages	<ol style="list-style-type: none"> <li>1. Simply sorting by human vision.</li> <li>2. Can separate different type of metal.</li> </ol>	<p>Very robust and insensitivity to the noise of the signal.</p>	<ol style="list-style-type: none"> <li>1. Have double confirmation using NIR and machine vision.</li> <li>2. High colour classification accuracy.</li> </ol>	<ol style="list-style-type: none"> <li>1. More accurate and reliable in detecting aluminium metal.</li> <li>2. Can differentiate other type of metal.</li> </ol>	<p>As an actuator to replace human to do pick and place task.</p>
Disadvantages	<ol style="list-style-type: none"> <li>1. High cost to pay for labourers.</li> <li>2. Risk the health of laborers for working in extreme environment.</li> </ol>	<ol style="list-style-type: none"> <li>1. Less accurate due to it based on a simple Euclidean distance.</li> <li>2. Cannot recognise overlapping object.</li> </ol>	<ol style="list-style-type: none"> <li>1. Slow system since the principle component analysis is added to do the adjustment.</li> <li>2. Cannot recognise overlapping object.</li> </ol>	<ol style="list-style-type: none"> <li>1. Complex mechanism should be set up.</li> <li>2. Complex analysis and program needed.</li> </ol>	<ol style="list-style-type: none"> <li>1. Need controller to assign task to it.</li> <li>2. Hard to design feedback loop to have better accuracy.</li> </ol>

## 2.3 Image processing technique

Digital image processing was the main tool to design a mechanism relay on machine vision. The following section introduced the idea of it and shown its function.

### 2.3.1 Introduction

The fundamental steps involved in digital image processing can be divided into two board of categories, which included method whose input and output are image and method that get attribute as output from the input image [15].

Image Acquisition was the first step to process image. The image capture from real world was converted into digital form and ready to be undergone per-processing like scaling the size of image.

The second stage is Image Enhancement. The role of this stage was to bring out the detail of the image or to emphasize on the feature of interest of the image. Furthermore, it also removed noise and made image more visually appealing. Enhancement could be very subjective region for image processing, it depended on which part the user would like to enhance. After that, Image Restoration took part to improve the appearance of an image which refer to mathematical or probabilistic model of image degradation [15].

Colour Image Processing extracted feature of interest from an image. Image Compression be a step to compress the storage size of image without affecting the quality to maximize the storage in camera or computer.

Morphological processing represented and described shape. This step was the key point to provide the output as attribute. Step in Segmentation was to partition an image into its constituent part. The more accurate the segmentation, the more likely for object recognition by using machine vision.

The last part was Representation and Description. It used the output from segmentation (raw pixel data) for computer to process and to fulfil the task of machine

vision. Hence, there are 9 processes involved in digital image processing as shown in Figure 2.7.

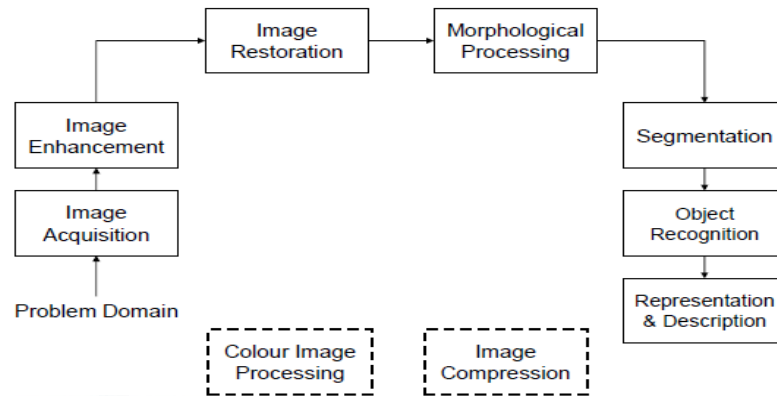


Figure 2.7: Nine key stages in digital image processing [15]

### 2.3.2 Histogram in image processing

In the world of image processing, 0 denoted as black and 255 denoted as white. The range stayed between 0 until 255, the higher and value the brighter the pixel. It was very helpful in image enhancement by using histogram processing. Histogram of an image shown the distribution of grey level in the image. A dark image had components of the histogram on the low side, which neared 0, while the brighter image was biased toward 255 regions. An image with low contrast tended to have narrow and middle concentrated histogram. The simple way to improve dark and washed out image is using histogram equalisation. Spread out frequency in an image will turn to high contrast image. Figure 2.8 illustrated image enhancement by using histogram equalisation. The formula for histogram equalisation was given by:

$$S_k = T(r_k)$$

$$\begin{aligned}
 &= \sum_{j=0}^k P_r(r_j) \\
 &= \sum_{j=0}^k \frac{n_j}{n} \quad (2)
 \end{aligned}$$

where  $r_k$  is input intensity

$S_k$  is processed intensity

$k$  is the intensity range

$n_j$  is the frequency of intensity  $j$

$n$  is the sum of all frequencies



Figure 2.8: Image enhancement by using histogram equalisation [15]

Plastic recycling by using the histogram of intensity technique to differentiate PET and Non-PET based to the property of transparency and opacity [16]. The steps were taken to sort the type of plastic included pre-processing, feature extraction and classification. From the pre-processing, the steps involved image resizing, filtering and getting silhouette image to reduce the noise due to lighting and background subtraction. The final step in pre-

processing is region properties measurement, the aim is to return the image in a structure array. When coming to feature extraction, the decision-making process was speeded and facilitated by representing the image in its decreased and compact form. This consisted of two algorithms, the first algorithm was studying the histogram of intensity from the whole image while the second one was to segment the image into five regions as shown in Figure 2.9, allowed the fifth region became the region of interest (ROI). Bounding box (BB) image algorithm was added to get an average of white pixel value from the grey scale plastic bottle image to get the representation of the smallest rectangle that can contain a region as shown in Figure 2.10. It could be concluded that if the bottle was not in upright position and the bottle label is not fixed in the middle, the system will be degraded. Figure 2.11 illustrated block diagram of plastic bottle classification.



Figure 2.9: All five segmented regions and generated ROIs [16]

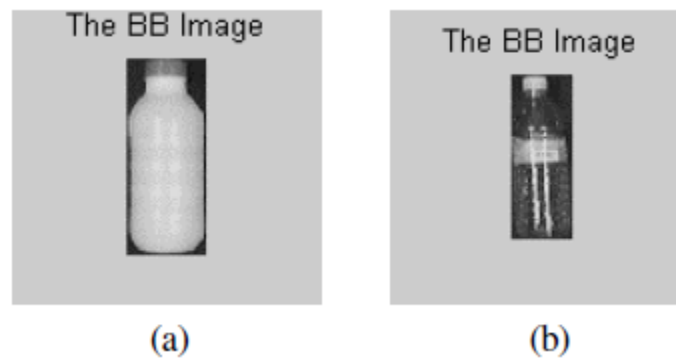


Figure 2.10: Bounding box image [16]

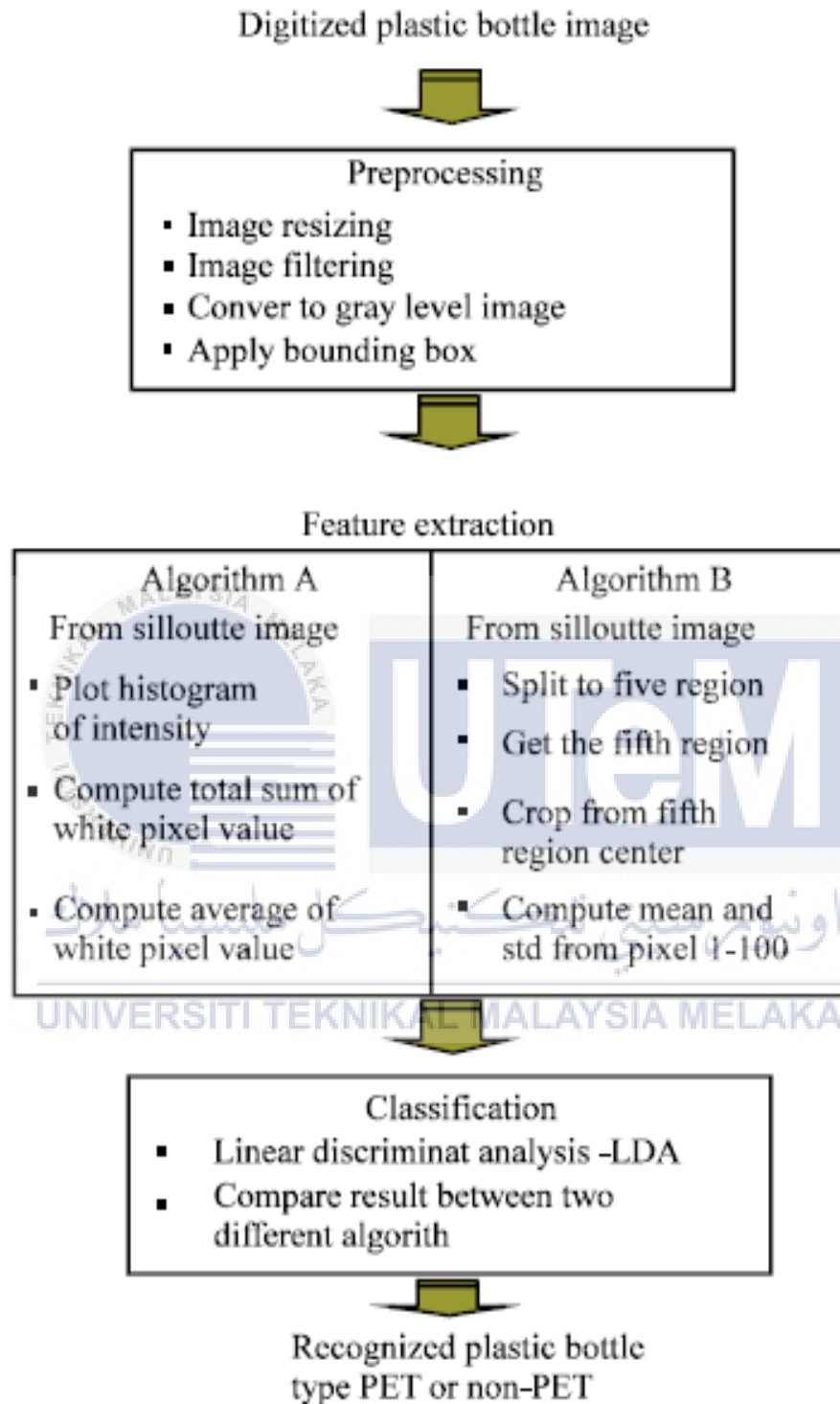


Figure 2.11: Block diagram of plastic bottle classification [16]

### 2.3.3 Logic Operation and Morphological in image processing

Arithmetic/Logic operation used two or more image to perform enhancement on pixel by pixel except NOT operation which just need single image. NOT operation was known as negative transformation. AND operation and OR operation were using original image to get the region of interest as shown in Figure 2.12 and Figure 2.13.

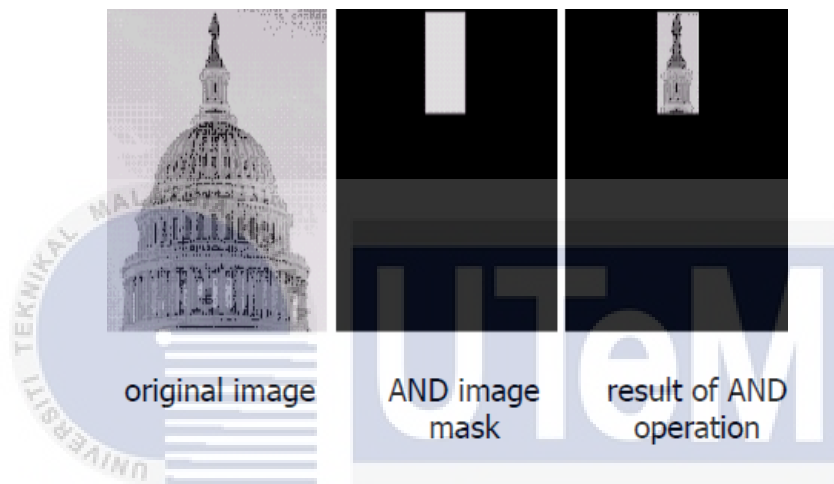


Figure 2.12: Image enhancement using AND operator [17]

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

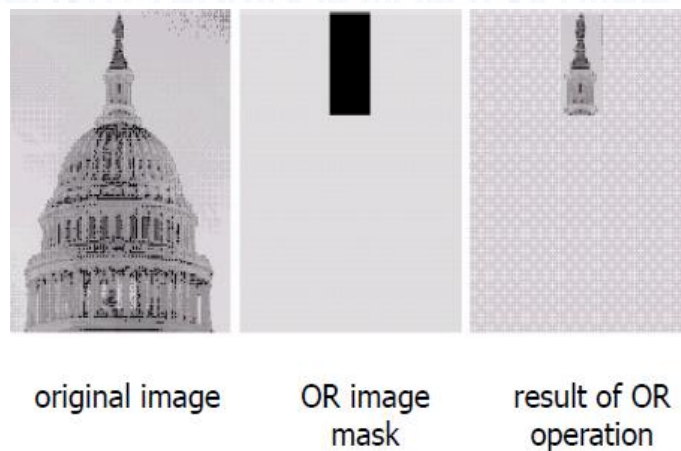


Figure 2.13: Image enhancement using OR operator [17]



Morphological was defined as a branch of biology that relate with the structure of animals and plants. In machine vision, the same word was used in the context of morphology mathematical to extract image components. These components allowed to descript and represent interest of region shape like boundary. One of the basic morphological algorithm was boundary extraction as shown in Figure 2.14 while Figure 2.15 below shown an example result of boundary extraction form binary image by using pervious algorithm. A boundary of set A was denoted by  $\beta(A)$  was equalled to:

$$\beta(A) = A - (A \ominus B) \quad (3)$$

Where: A = Set A

B = Structuring Element B

$A \ominus B$  =, A eroded by B

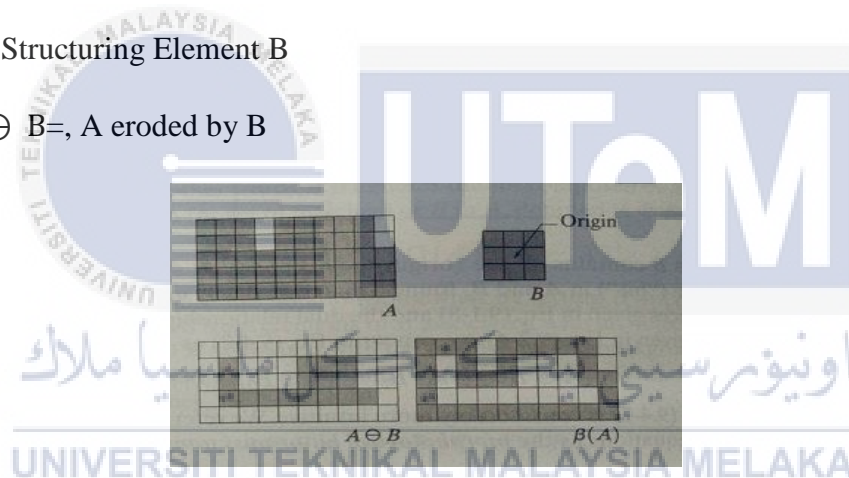


Figure 2.14: Set A, Structuring Element B, A eroded by B and Boundary machines of boundary extraction [17]

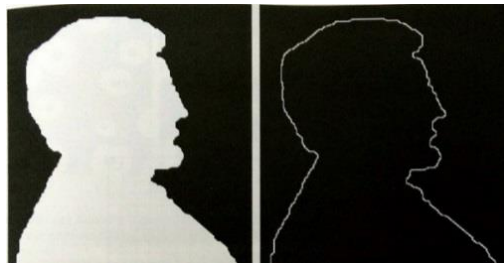


Figure 2.15: Result of using boundary extraction algorithm from binary image [17]

### 2.3.4 Segmentation in image processing

Segmentation was a major process to get an image attributes. This process subdivides an image into its constituent region or object. Segmentation was used to detect object or region of interest in an application. If the region of interest was isolated, that segmentation should be stopped. Theoretically, segmentation involved two basic properties, which included discontinuity and similarity. Discontinuity was an approach to separate an image according to abrupt change in intensity like object edges in an image. Similarities was an approach to partition image into regions that were similar based on a set of predefined rules like thresholding and merging [17].

The most common way to detect discontinuity was to run a 3x3 “mask” through an image as shown in Figure 2.16 below. The response of the mas at any point in the image will be:

$$R = w_1z_1 + w_2z_2 + \dots + w_9z_9 = \sum_{i=1}^9 w_i z_i \quad (4)$$

Where,  $z_i$  is the grey level of the pixel associated with mask coefficient  $w_i$ .

$w_1$	$w_2$	$w_3$
$w_4$	$w_5$	$w_6$
$w_7$	$w_8$	$w_9$

Figure 2.16: 3x3 mask [15]

Machine vision was employing the mask above for line detection. The mask will move horizontally around the image. Ideal result will be appealed when the line passed through the middle row of the mask. There are 4 types of line detection mask, each of them have different orientation like horizontal, +45, -45° and vertical. In term of selecting the

suitable mask for line detection, Equation 8 is used to get the orientation value. The highest gain is then chosen.

A robust technique for the automatic segmentation and classification of touching plastic bottles was introduced in [18]. It can have two or more input (plastic bottle) at one time without considering whether they are touching each other or not as shown in Figure 2.17. This technique used genetic approach to determine the position of bottles and differentiate the two objects by means of a segment of straight line. The algorithms used came from Darwinian Theory of transformation of a population of individual objects into new generations to keep the fittest one. It was an efficient optimisation tool since it solved the cluster separation problem. It was selected to solve the bottle overlapping problem. Based on the technique, fifty lines were built between two bottles based on the heuristic rules. They were connected in random couple of perimeter picture element. After that, the 10 fittest individuals are kept as parents for new generations. In the result of segmentation, a line is selected to be the best of fitting element as shown in Figure 2.18.

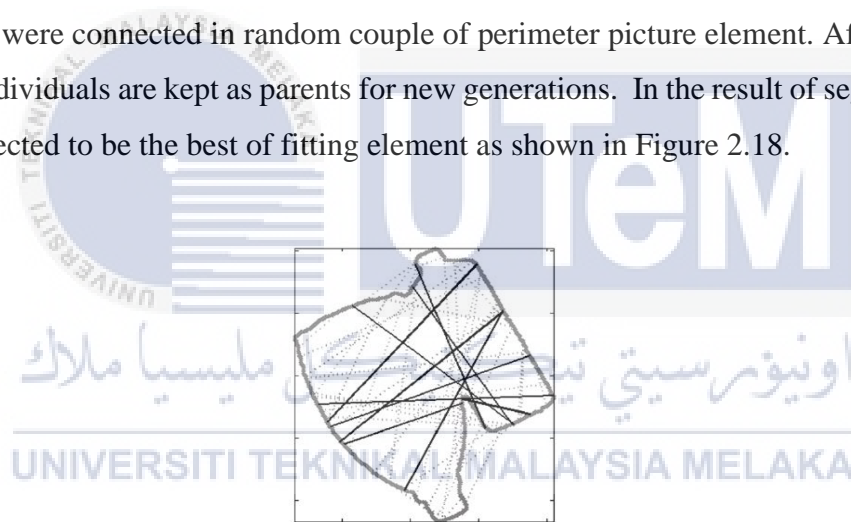


Figure 2.17: Initial population of 50 lines [18]

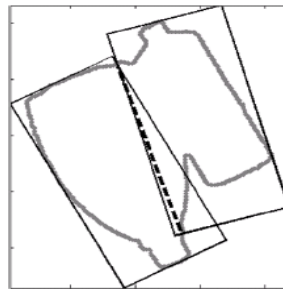


Figure 2.18: Final population and the thick black line (best-fitting element) [18]

### 2.3.5 Colour in image processing

Sometimes it was much easier to detect the object identification based on object colour. Colour was also a common factor to perform image analysis. Colour image processing was categorised into two major areas that is Full-colour and Pseudo code colour processing. The usage of Full-colour normally apply in colour TV camera and colour scanner while Pseudo code colour processing was to assign a colour to a range of interest.

The characteristics to distinguish a colour from another Brightness, Hue and Saturation. Brightness was defined as chromatic notion of intensity. Hue was known as dominant colour as perceived from input. Saturation shown the relative purity or the amount of white light mixed with hue. For example, pure spectrum colour were fully saturated while lavender colour was referred as less saturated (mixed colour) [17].

Hue and saturation mixed together were known as chromaticity. Hence, colour could be categorised by just brightness and chromaticity. The total value of red, blue and green could be calculated and shown by using equation:

$$X = \frac{x}{x+y+z} \quad (5)$$

$$Y = \frac{y}{x+y+z} \quad (6)$$

$$Z = \frac{z}{x+y+z} \quad (7)$$

Where, X, Y and Z were tristimulus values. The summation of X, Y and Z should be equal to 1.

There was some theory and calculation for the conversion colour form RGB to HSI. Considering the image is in RGB colour format, then Hue (H) component of every pixel was calculated by using equation:

$$H = \begin{cases} \theta & \text{if } B \leq G \\ 360^\circ - \theta & \text{if } B > G \end{cases} \quad (8)$$

Where

$$\theta = \cos^{-1} \left\{ \frac{0.5[(R-G)+(R-B)]}{[(R-G)^2+(R-B)(G-B)]^{0.5}} \right\} \quad (9)$$

Saturation components:

$$S = 1 - \frac{3}{(R+B+G)} [\min(R, G, B)] \quad (10)$$

Intensity Components:

$$I = \frac{1}{3}(R + G + B) \quad (11)$$

RGB values is in range of [0,1] and the angle  $\theta$  was measured with the respect of red axis of HSI space as shown in Figure 2.19.

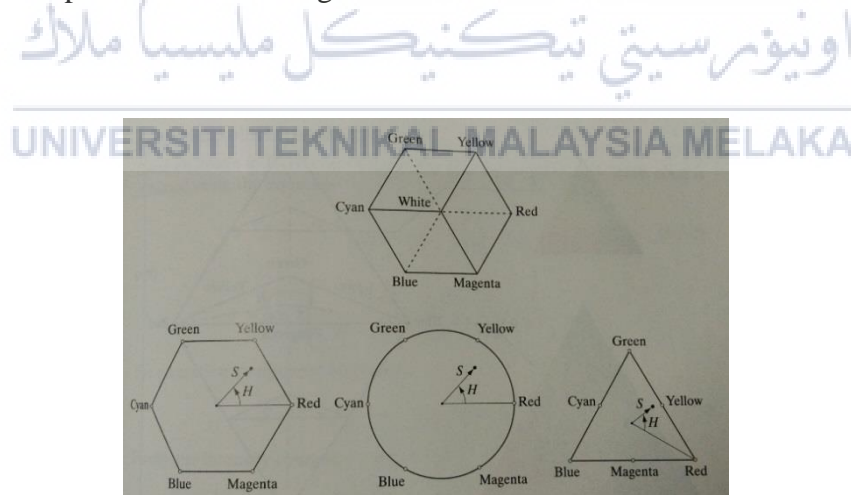


Figure 2.19: Hue and saturation in the HSI colour model [17]

Colour classification using fuzzy inference and genetic algorithm to discriminate the various type of component was proposed [19]. This method helped to speed up the

process of classification of objects. When facing unknown colour, it allowed to utilize fuzzy inference to category it into one of the reference colour. Fuzzy inference was optimized by added genetic algorithm. The general process of colour classification using fuzzy inference included light sources for illumination, RGB colour sensor to detect Red, Blue, and Green and some parameters which stated in Figure 2.20, where r = Red, g = Green, b = Blue, and s = standard white sample. V is the output value. This method was applied to classify glass bottles due to the colour of bottle were inconsistent, so made it possible for recycling the glass bottles in real world by using machine vision. Figure 2.21 shown chromaticity graph of different type of glass bottles. The five-important equation of fuzzy were listed below:

$$V_s = \frac{V_r}{V_{r0}} + \frac{V_g}{V_{g0}} + \frac{V_b}{V_{b0}} \quad (12)$$

$$r = \frac{(V_r/V_{r0})}{V_s} \quad (13)$$

$$g = \frac{(V_g/V_{g0})}{V_s} \quad (14)$$

$$b = \frac{(V_b/V_{b0})}{V_s} \quad (15)$$

$$s = \frac{V_s}{V_{s0}} \quad (16)$$

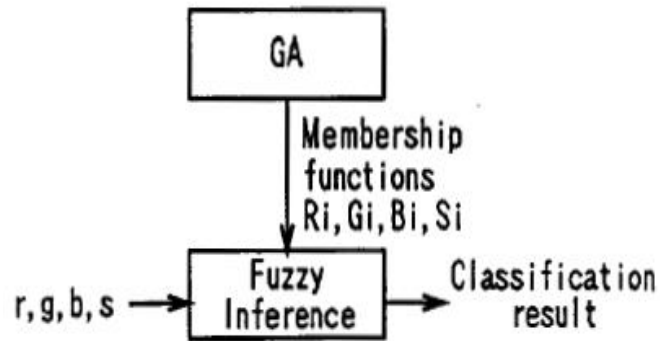


Figure 2.20: Concept of colour classification by using fuzzy and genetic algorithm [19]

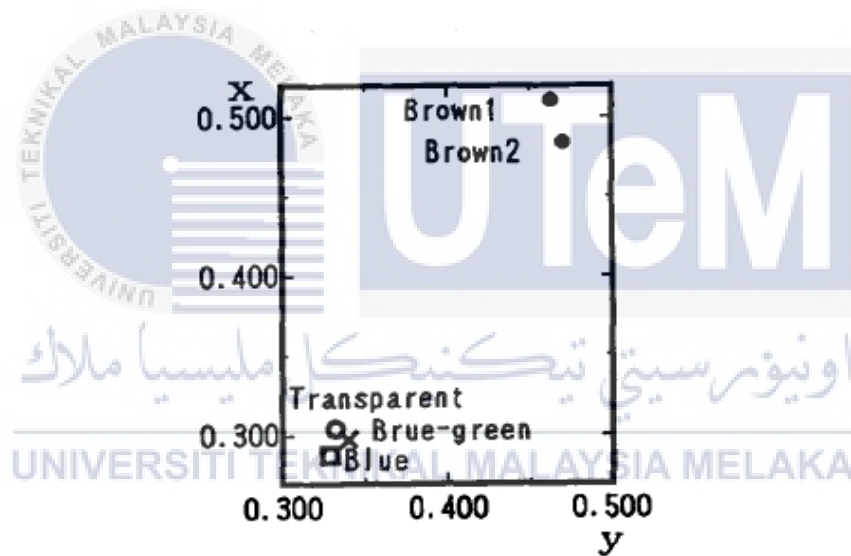


Figure 2.21: The chromaticity graph of different type of glass bottles [19]

### 2.3.6 Decision Based on Neighbourhood White Strip

A “Probabilistic White Strip Approach to Plastic Bottle Sorting System” by using a decision based on the neighbourhood information of the white strip was developed [12]. Figure 2.22 shown the flow of plastic sorting. The algorithm built assumed that for any detected white strip, the length of white strip will be mostly grey if it was parallel with the neighbourhood colour information. The step for the sorting process started from input image, then foreground mask was constructed to separate the bottles and the conveyor belt as shown in Figure 2.23. In this part, it involved subtraction collection of connected pixel of the foreground. After that, white strip was used to investigate the entire image by assuming 3 dimensional exponential distributions of the RGB (Red, Green, and Blue) channels. Contour box was then built for each detected white strip, and 8-neighbourhood (4-neighbourhood with diagonal-neighbourhood, was a tool of distance measure) in Figure 2.24 was built around the strip. Turning to the next step, a posterior was maximised before built histogram of small sub-region. The selected algorithms as the benchmark for performance comparison were from [16] and [17]. The result shown that the algorithm by [20] is the best among six of the total algorithm, with the accuracy of 80.7% to differentiate PET from Non-PET.

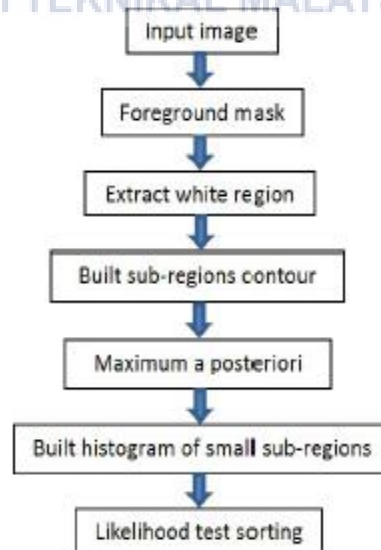


Figure 2.22: Flow of plastic sorting [20]



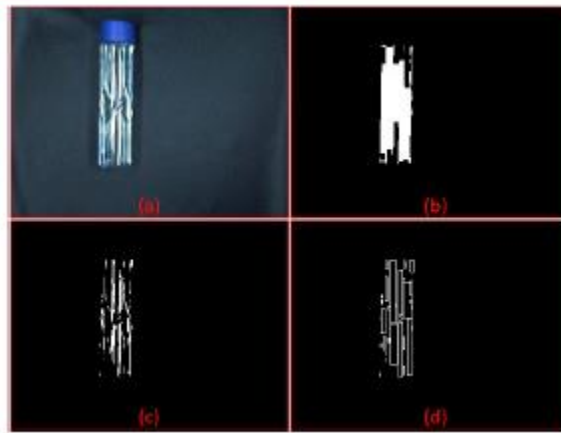


Figure 2.23: (a) Input image, (b) Extracted foreground, (c) White strips, (d) Contour boxes [20]

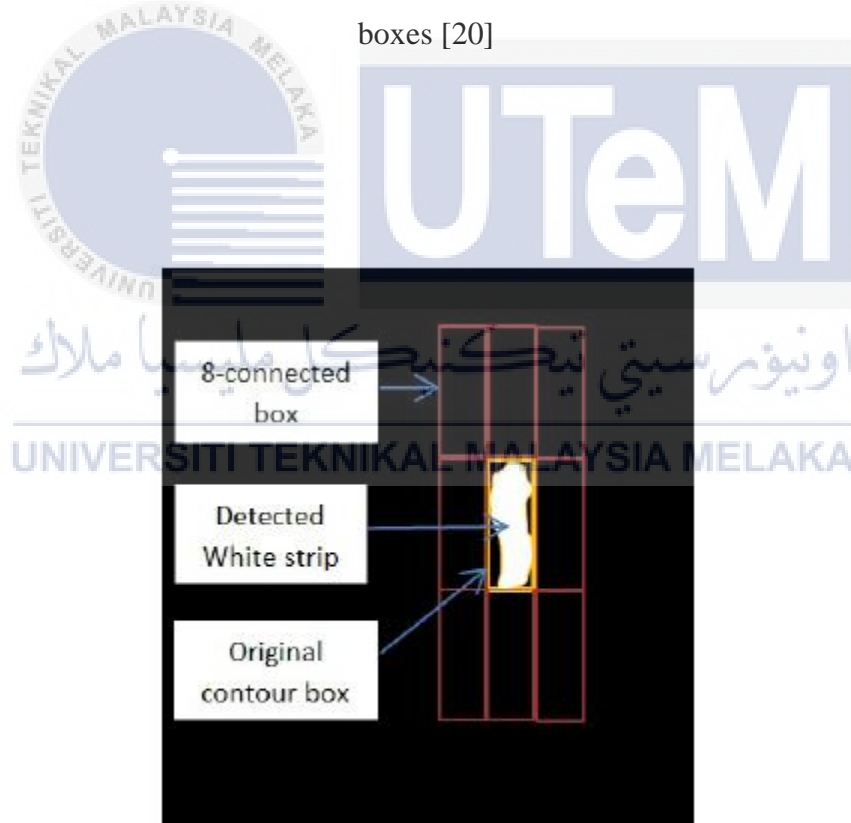


Figure 2.24: 8-connected box for each white strip [20]

### 2.3.7 Object Shape Recognition

A shape recognition method to detect objects on conveyor by using easy understanding and low computation algorithm was introduced in [21]. From the input image, this method extracted its intensity value, which then used Otsu's method for threshold. After that, the binary image was obtained. The proposed method using Otsu's method for thresholding probably due to it able to automatic choose the threshold from the grayscale histogram to make the threshold image categorised as foreground and background. Sobel operator was used to find the edge of object for future used in object recognition. Sometime, some unwanted edges might be counted in parameter algorithm. The approach to eliminate this phenomenon was applying Thinning method. Thus, error detection could be reduced. Noise appeared in the image processing was filtered by median filtering. Compactness calculation was taken to differentiate the shape between object. This approach was emphasized on the shape of circle, triangle and square as shown in Figure 2.25. It able to achieve the successful rate at 85% of shape detection. However, the weakness mentioned included this approach was sensitive to noise and light condition. Improvement in lighting condition could increase the successively detection.

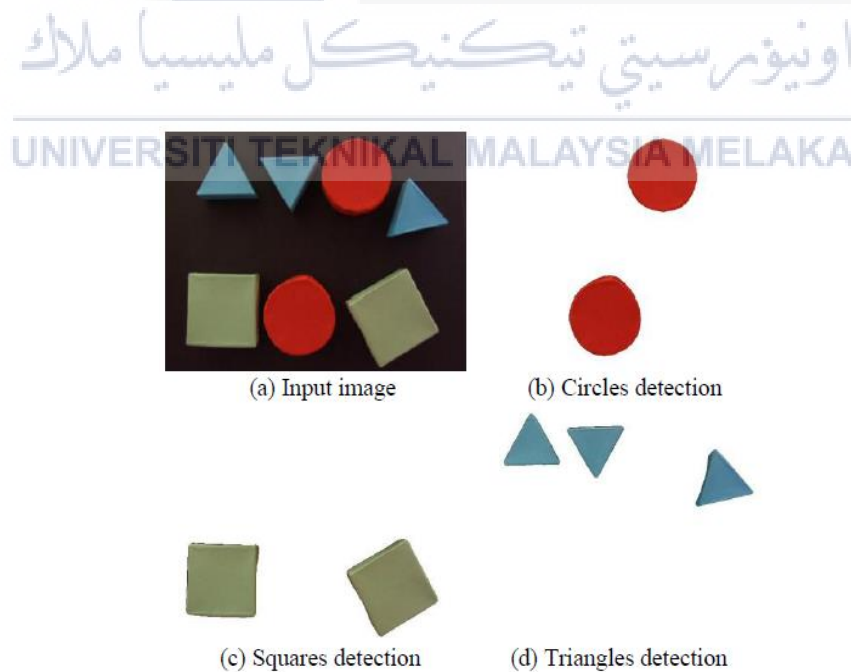


Figure 2.25: Example of shape detection [21]

Table 2.3: Image processing technique overview.

	Histogram in image processing [16]	Logic Operation and morphological in image processing [17]	Segmentation and Classification Method [18]	Colour in image processing [19]	Decision Based on Neighbourhood White Strip [20]	Object Shape Recognition [21]
Properties	<ol style="list-style-type: none"> <li>1. Extract the property of transparency and opacity.</li> <li>2. Show the distribution of grey level in an image.</li> </ol>	<ol style="list-style-type: none"> <li>1. Extract region of interest in an image.</li> <li>2. Descript and represent interest of region shape like boundary</li> </ol>	Segmentation and Classification	Detect the object identification based on object colour	8 neighbour-hood distance measure	<ol style="list-style-type: none"> <li>1. Otsu's method for threshold</li> <li>2. Sobel operator for extract edge</li> <li>3. Compactness calculation for categorise</li> </ol>
Advantages	Double confirmation by using histogram of intensity from the whole	Reduce noise and eliminate unwanted region.	1. Can have two or more input at one time.	1. Unknown colour object detected will category into one of the	Have highest accuracy of 80.7% to differentiate PET from Non-PET	<ol style="list-style-type: none"> <li>1. Easy understanding.</li> <li>2. Low computation algorithm.</li> </ol>

	image and segment the image to focus on interest part.		2. Solve object overlapping problem.	reference colour.  2. Speed up the process of separation	compare with other in this table.	3. Can have two or more object at a time.
Disadvantages	1. Not stable when the bottle is not in upright position and the bottle label is not fixed in the middle. 2. Cannot recognise overlapping object.	Important part may be eliminated.	Cannot differentiate PET or Non-PET.	1. Sensitive to light intensity and light reflection. 2. Cannot recognise shape.	1. Complicated coding and high resolution camera are needed. 2. Cannot recognise overlapping object.	1. Less accurate to recognise complex shape. 2. Cannot recognise overlapping object. 3. Cannot differentiate PET or Non-PET.

## 2.4 Summary of chapter

As a summary from this literature review, recycle is the only way to separate waste accordingly. Machine vision able to replace human eye to separate used drinking bottles or cans based on its material like plastic, aluminium and glass. All these materials have its own unique reflective of light and colour. These parameters are the key point to develop algorithm in next chapter. After having useful parameter, some important techniques in image digital processing are needed. In next chapter, Colour Processing and Histogram Equalization are used in digital image processing. Pick and place robot in Section 2.2.5 shown a close loop system. In this project, an open loop system is chosen and implemented since the requirement only need to transfer target from basement to three different location. Last but not least, these reviews are the fundamental and guiding line for this project to run smoothly.



## CHAPTER 3

### METHODOLOGY

The design of Automated Waste Separated Machine was divided into 5 major section, which included analysis, hardware, coding, electric circuit and testing section.

#### 3.1 Determine possible shape, perimeter and colour of bottles.

The analysis is done by referring to the literature review in Chapter 2. Waste can be categorised in recycle, kitchen, hospital (harmful) and other waste. Hence, waste materials like plastic bottles, aluminium cans and glass bottles are aimed for this project to automatically sort them. Table 2.2 (Mechanical technique overview) and Table 3.3 (Image processing technique overview) are shown to compare the waste separate method based on waste characteristic.

There are various kind of appearance for each type of waste, thus, after making analysis, this project is built to sort waste bottle or cans based on colour and reflective of light by using algorithm, histogram equalization and some calculation. Every conformation of the wastes is needed to recognize well through the programming system based on the image captured [17].

### 3.2 Hardware development

Three major parts are focused in this hardware development:

- SolidWork is used to design the outlook of system.
- Controller is selected to provide output based on scanning result.
- Suitable camera is chosen to capture image.

First of all, Solidwork is used to draft the outlook of half cover black box as shown in Appendix section 7.1. Assembly technique is applied to combine all the parts. After that, bill of material (BOM) is used to indicate the materials needed of the real hardware. For prototyping, the materials used are based on low budget.

Suitable size of gripper, servomotors MG995R and MG 996R are purchased through online shop to save time and budget. Servomotor act as an actuator to control grip, release and direction of gripper.

Next, Arduino Uno is selected to be the controller of the system due to it is using C++ language and can pair with openCV. This controller allows to synchronize with machine vision (input) to provide output to actuate gripper to perform pick and place function. Furthermore, there are a lot of sources provided in internet as reference. An example is shared in open sources using openFrameworks pair with Arduino and Servomotor to perform object tracking system [22]. Another example of detecting and tracking face function available in [23] which also using Arduino and openCV. All these examples show that Arduino Uno is a very powerful controller.

Brand of Jingui Web camera is chosen for this project due to its high resolution with 12MPixel, built it On/Off LED, adjustable focus length, automatic low-light correction to reduce the noise and improve image quality, hi-Speed USB 2.0 to fast data transfer and tripod-ready universal clip fits laptops monitors [24].

### 3.3 Software Overview and Selection

When come to software part there a lot of software can be used for machine vision. Machine vision is a field of science, which able to replace human eyesight to recognize objects in real world. This technique is come from digital images or videos using sophisticated methods and transform it to other representation, the fundamental example let it to be binary objects. From the following research, the best recognition system is chosen for this FYP as a suitable medium to type command.

#### 3.3.1 VXL

VXL, also known as Vision-Something-Libraries are the computer vision C++ based libraries. It focusses on light, fast and consistent system. Tone of its strange is portable over many platforms. Core of VXL is divided into few parts like numeric (vnl), imaging (vil), geometry (vgl), streaming (vsl), basic templates (vbl), utilities (vul) and so on. X is stand for the variable used. Moreover, it is self-independent. They could be used separately [25].

#### 3.3.2 LTI

LTI is based on object oriented primarily used in image processing and computer vision. It is made on Aachen University of Technology as a part of other computer and robotics research projects. The main idea of this library is to simplify the code sharing and maintenance as much as is possible. It can be used in both Linux and Windows operation computer. The main parts are Linear algebra, Classification and Clustering, Image processing and Visualization and Drawing Tools. The disadvantage is this library is usable under GNU license [26].



### 3.3.3 OpenCV

OpenCV is also known as “Open Source Computer Vision”, is a software library. It provides high level computer vision and machine learning functionalities. OpenCV not only provides a common infrastructure for computer vision applications but also to decrease the software developing time. It is so different and suitable under-graduate student because it guarantees free to be used. Besides that, it has about 2500 optimized algorithms. Algorithms are usable in face detection and recognition, object recognition, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements and recognize scenery and establish markers to overlay it with augmented reality.

This kind of library has been used in Google StreetView images, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, detection of Europe swimming pool drowning accidents, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan and more other. They are 5 interfaces for work with library C++, C, Python, Java and MATLAB. Supported are the most used operation systems like Windows, Linux, Android and MAC [27].

A lot of sources provide lessons and tutorials about machine vision by using OpenCV. Image processing means a way to manipulate image data so that the image can be used for computer vision application [28]. The most common usage of application is object recognition and decision making just based on input image and algorithm used in the software. Simple explanation about installing and configuring OpenCV in Microsoft visual studio were available online. Besides that, the ways to use OpenCV to read and show image, filtering image by changing brightness, contrast, and using histogram equalization are illustrated. Moving to higher level, detecting object by using camera became possible. OpenCV include library to detect colour and shape, basic steps must be gone through to provide object tracking and recognition.

Table 3.1: Comparison between different types of machine vision software

No	Software Criteria	VXL [25]	LTI [26]	OpenCV [28]
1.	Supported Operation System	<ol style="list-style-type: none"> <li>1. Linux</li> <li>2. Windows</li> <li>3. BSD</li> </ol>	<ol style="list-style-type: none"> <li>1. Linux</li> <li>2. Windows</li> </ol>	<ol style="list-style-type: none"> <li>1. Linux</li> <li>2. Windows</li> <li>3. Android</li> <li>4. MAC</li> </ol>
2.	Interfaces for work with library	<ol style="list-style-type: none"> <li>C++</li> <li>C</li> </ol>	<ol style="list-style-type: none"> <li>C++</li> <li>C</li> </ol>	<ol style="list-style-type: none"> <li>1. C++</li> <li>2. C</li> <li>3. Python</li> <li>4. Java</li> <li>5. MATLAB</li> </ol>
3.	Feature	<ol style="list-style-type: none"> <li>1. VNL (numeric): Numerical containers and algorithms.</li> <li>2. VIL (imaging): Loading, saving and manipulating images in many</li> </ol>	<ol style="list-style-type: none"> <li>1. Extract eigenvalues, eigenvectors, linear equations solutions, statistics.</li> <li>2. Classification and Clustering Radial Basis Function classifiers</li> </ol>	<ol style="list-style-type: none"> <li>1. Able to adjust variety of images (image processing).</li> <li>2. Identifying objects in the image (especially faces algorithm improved Viola Jones)</li> </ol>

		common file formats, including very large images. 3. VGL (geometry): Geometry for points, curves and other elementary objects in 1, 2 or 3 dimensions. 4. VSL (streaming I/O) 5. VBL (basic templates) 6. VUL (utilities): Miscellaneous	3. Segmentation approaches, linear filters, wavelets, steerable filters 4. Visualization and Drawing to save lots of time in image programming	3. Traffic tracking 4. Identifying human indicators.
4.	Source Type	Open	Usable under GNU license	Open
5.	Pricing	Free	Not Stated	Free

In conclusion, OpenCV is chosen for this Final Year Project with the title of Automated Waste Separated Machine. The decision is due to it is supported by 4 differences operation system, causes it has thousands more of resources in internet. Besides that, it is free to use and very suitable for students that have basic knowledge in C++ programming. Furthermore, the requirement of this project just involves colour processing to do object recognition, OpenCV library is added in Visual Studio fulfilled the requirement.

### 3.4 Write coding for comparison and actuation between capture image and database image

An algorithm will be designed that when a waste is placed in camera range, the image will be captured and send to laptop then converted from true colour image to HSV (Hue, Saturated and Value) image and finally the image with free pixel. After the image changed its hue and saturation, the edited image is sent to processor for comparison between image and programming. The processor will identify the image information and classify into few categories. This section not only focus on coding development by using OpenCV and C++ programming, but also Arduino coding to control actuator by receiving output command from Visual Studio through serial communication. There are three part of it to run the job:

1. Use the image captured to determine properties of waste.
2. From comparison between the images, recognise the object or waste, then give output to actuator to sort it.
3. Use the properties of waste to let pick and place gripper transfer it to next station.

Coding of image processing in C++ and output command in Arduino language were illustrated in Appendix Section 7.5.and Section 7.6

### 3.5 Proof of concept by using Visual Studio C++ and Arduino

After write down the coding for image processing and actuation command, testing is performed with Visual Studio and Arduino platform. For example, input two same images contained single bottle, then run the program to do comparison. If both images are the same, the comparison value should be 1 (means 100% identical). If the value does not meet value 1, troubleshooting must be carried out. After proving reliability of image processing, serial communication between Visual Studio and Arduino Uno is tested by lighting up Arduino LED pin 13.

### 3.6 Electric circuit development

It acts as a connector between hardware and software. In this section, testing board is used to design and test electrical circuits. The circuit design starts from power supply to power up micro-processor (Arduino Uno), motor, actuator and camera. By using it, prototype iterations can be minimized to reduce budget and time. In addition, the instruments that provided by faculty will not get burn easily.

### 3.7 Development of full structure of mechanism

Arduino and servomotors will take part as mechanical controller to control of gripper. In this full structure, there are 3 bins to store different types of bottle, which are glass, plastic and aluminium respectively. The role of gripper was to transfer the bottle into specific bin based on its material property. This section is the combination of hardware, coding and circuit that developed in pervious sections. Figure 3.1, Figure 3.2 and Figure 3.3 shown the full structure of mechanism in different views.



Figure 3.1 : Side view

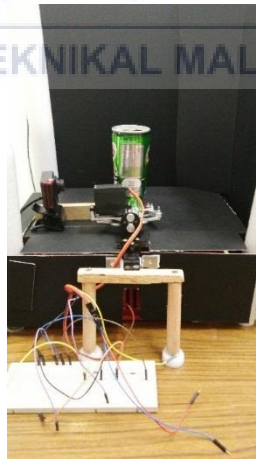


Figure 3.2 : Top view



Figure 3.3 : Front view

### 3.8 Proof of the Machine Vision with full structure of actuator

Cooperation between actuator system and machine vision system is needed to sort waste automatically. First, camera will always capture image and sent it to laptop. The duration to capture image is set at every 1 second if no object detected. Image processing will be done in the laptop to differentiate the type of used drinking container. When it is tally with the analysis result, the programming will instruct gripper to pick up the waste, turn to certain angle and transfer it to desire destination. LED light was lighted up at the same time. Figure 3.4 shown the first test of functionality of mechanism.

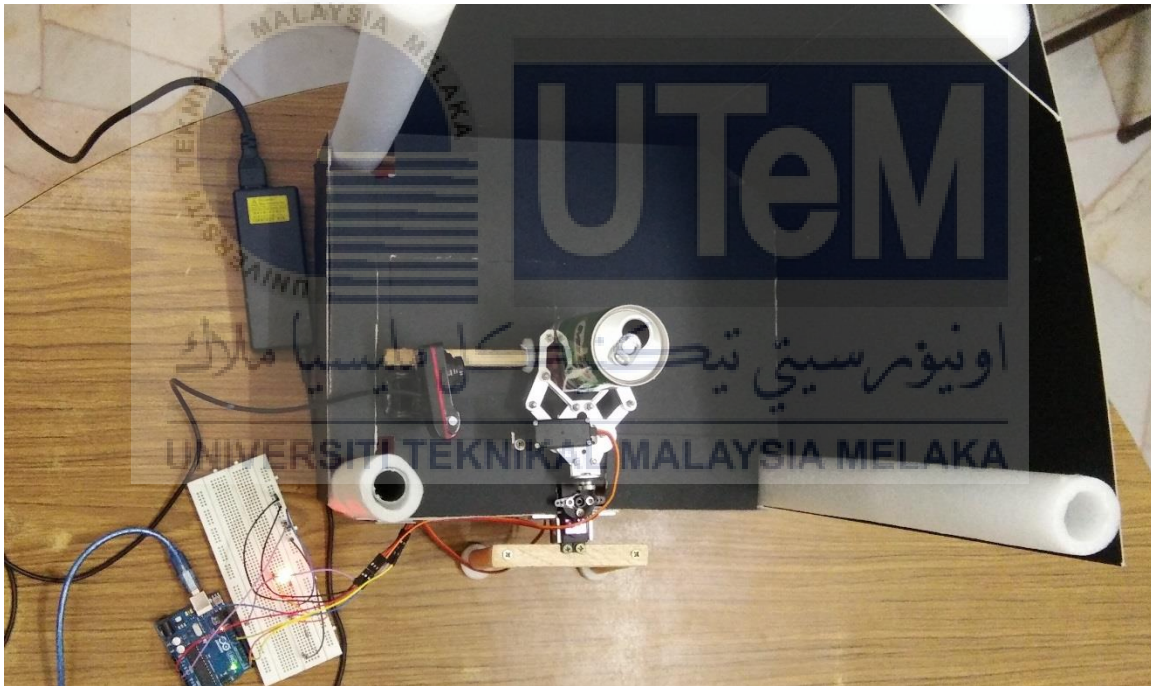


Figure 3.4: Functionality testing of mechanism with LED light up



### 3.9 Experiment

There were four type of experiments to carry out for testing overall machine reliability and accuracy. Each experiment consisted of three set for 10 trial. Accuracy value will be calculated based on formula from equation 17 to 33. Every result of comparison value would be shown Window Console and saved in note for data analysis as shown in Figure 3.5.

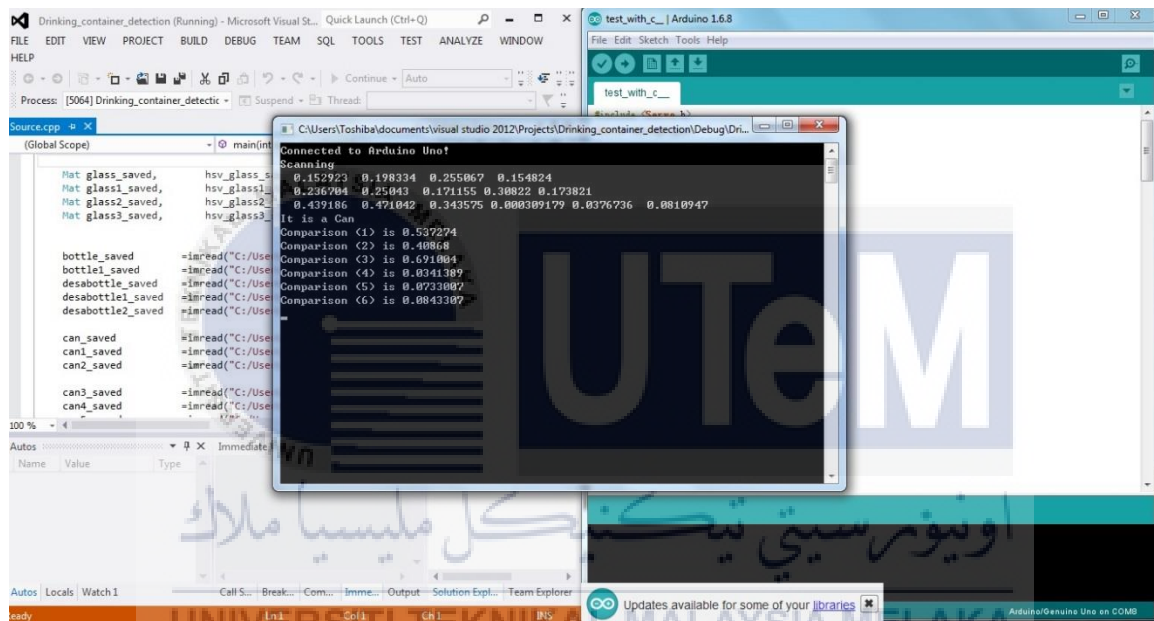


Figure 3.5: Console showed value of comparison of aluminium can

#### 1. Testing for machine vision with 30 input of plastic bottles

$$a. \text{ Accuracy (1.1)} = 1 - \frac{\text{number of miss identify plastic bottle}}{10} \quad (17)$$

$$b. \text{ Accuracy (1.2)} = 1 - \frac{\text{number of miss identify plastic bottle}}{10} \quad (18)$$

$$c. \text{ Accuracy (1.3)} = 1 - \frac{\text{number of miss identify plastic bottle}}{10} \quad (19)$$

$$d. \text{ Accuracy (average)} = \frac{\text{Accuracy (1.1)} + \text{Accuracy (1.2)} + \text{Accuracy (1.3)}}{3} \quad (20)$$

## 2. Testing for machine vision 30 input of glass bottles

$$\text{a. Accuracy (2.1)} = 1 - \frac{\text{number of miss identify glass bottle}}{10} \quad (21)$$

$$\text{b. Accuracy (2.2)} = 1 - \frac{\text{number of miss identify glass bottle}}{10} \quad (22)$$

$$\text{c. Accuracy (2.3)} = 1 - \frac{\text{number of miss identify glass bottle}}{10} \quad (23)$$

$$\text{d. Accuracy (average)} = \frac{\text{Accuracy (2.1)} + \text{Accuracy (2.2)} + \text{Accuracy (2.3)}}{3} \quad (24)$$

## 3. Testing for machine vision 30 input of aluminium cans

$$\text{a. Accuracy (3.1)} = 1 - \frac{\text{number of miss identify aluminium}}{10} \quad (25)$$

$$\text{b. Accuracy (3.2)} = 1 - \frac{\text{number of miss identify aluminium}}{10} \quad (26)$$

$$\text{c. Accuracy (3.3)} = 1 - \frac{\text{number of miss identify aluminium}}{10} \quad (27)$$

$$\text{d. Accuracy (average)} = \frac{\text{Accuracy (3.1)} + \text{Accuracy (3.2)} + \text{Accuracy (3.3)}}{3} \quad (28)$$

## 4. Testing for machine vision 30 input of mixed drinking bottles

$$\text{a. Accuracy (4.1)} = 1 - \frac{\text{number of miss identify of mixed input}}{10} \quad (29)$$

$$\text{b. Accuracy (4.2)} = 1 - \frac{\text{number of miss identify of mixed input}}{10} \quad (30)$$

$$\text{c. Accuracy (4.3)} = 1 - \frac{\text{number of miss identify of mixed input}}{10} \quad (31)$$

$$\text{d. Accuracy (average)} = \frac{\text{Accuracy (4.1)} + \text{Accuracy (4.2)} + \text{Accuracy (4.3)}}{3} \quad (32)$$

After performed the four experiments shown in above, the accuracy need to reach 0.8 and above to achieve to show the whole machine is reliable to separate unbent waste bottles into categories.



### 3.10 Flow chart of Automated Waste Separated Machine

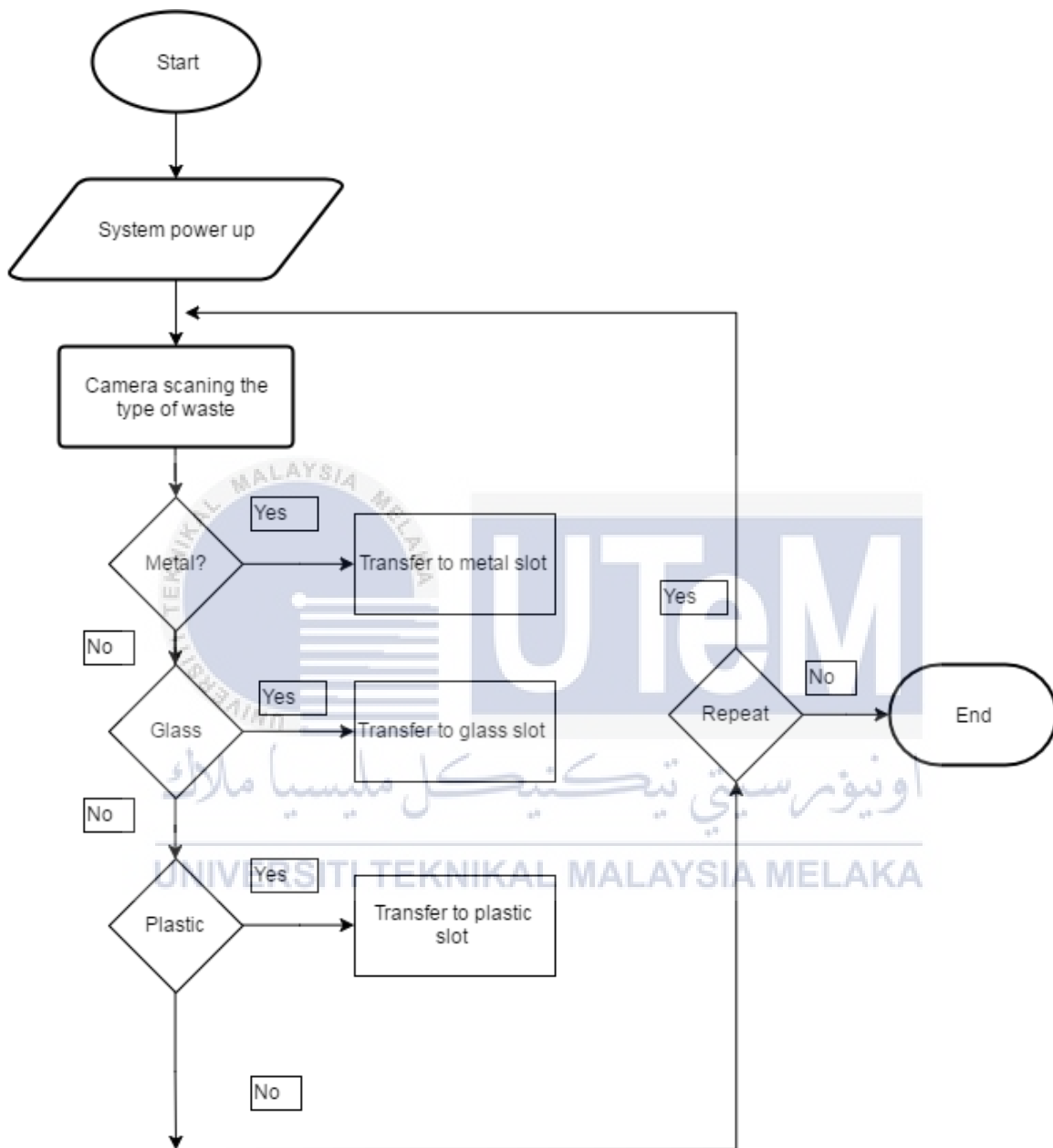


Figure 3.6: System flow chart

### 3.11 Flow of activities

This project took around 7 months to develop, which included FYP 1 and FYP 2. It was divided into 13 tasks. Time flow under blue colour is belongs to research and development categories, grey colour was installation and testing section, green colour was observation and analysis zone. After that, it will take around 4 weeks to make final improvement. Last but not least, document and report writing and final paper review need to took another one month to complete before submission. Gantt chart is shown in Appendix section 7.4.



## CHAPTER 4

### RESULT AND DISCUSSION

#### 4.1 Introduction of image processing result

In the following section, result is recorded to show the similarity between database image and real world captured image. The highest value (1) means that both images are 100% similar while lowest value (0) means that both images are totally different. The result consists of 4 subsections which included glass bottle accuracy testing in section 4.2.1, plastic bottle accuracy testing in section 4.2.3, aluminium can accuracy testing in section 4.2.5 and final testing for overall accuracy in section 4.2.7. Steps below shows the method used in C++ coding to differentiate type of drinking container.

1. Include serial communication command between Visual Studio and Arduino

```
HANDLE hSerial = CreateFile(L"COM8", GENERIC_READ | GENERIC_WRITE, 0, 0, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, 0);
if (hSerial !=INVALID_HANDLE_VALUE)
{
    printf("Connected to Arduino Uno! \n");

    DCB dcbSerialParams;
    GetCommState(hSerial,&dcbSerialParams);

    dcbSerialParams.BaudRate = CBR_9600;
    dcbSerialParams.ByteSize = 8;
    dcbSerialParams.Parity = NOPARITY;
    dcbSerialParams.StopBits = ONESTOPBIT;

    SetCommState(hSerial, &dcbSerialParams);
}
```

## 2. Assign name for images in database

```

Mat bottle_saved,      hsv_bottle_saved;
Mat bottle1_saved,    hsv_bottle1_saved;
Mat desabottle_saved, hsv_desabottle_saved;
Mat desabottle1_saved, hsv_desabottle1_saved;
Mat desabottle2_saved, hsv_desabottle2_saved;

Mat can_saved,        hsv_can_saved;
Mat can1_saved,      hsv_can1_saved;
Mat can2_saved,      hsv_can2_saved;

Mat can3_saved,      hsv_can3_saved;
Mat can4_saved,      hsv_can4_saved;
Mat can5_saved,      hsv_can5_saved;

Mat glass_saved,     hsv_glass_saved;
Mat glass1_saved,   hsv_glass1_saved;
Mat glass2_saved,   hsv_glass2_saved;
Mat glass3_saved,   hsv_glass3_saved;

```

## 3. Assign address for looking image in databased

```

bottle_saved      =imread("C:/Users/Toshiba/Pictures/opencv/bottle/bottle_1.jpg", CV_LOAD_IMAGE_COLOR);
bottle1_saved     =imread("C:/Users/Toshiba/Pictures/opencv/bottle/bottle_2.jpg", CV_LOAD_IMAGE_COLOR);
desabottle_saved  =imread("C:/Users/Toshiba/Pictures/opencv/desabottle/desabottle_1.jpg", CV_LOAD_IMAGE_COLOR);
desabottle1_saved =imread("C:/Users/Toshiba/Pictures/opencv/desabottle/desabottle_2.jpg", CV_LOAD_IMAGE_COLOR);
desabottle2_saved =imread("C:/Users/Toshiba/Pictures/opencv/desabottle/desabottle_3.jpg", CV_LOAD_IMAGE_COLOR);

```

## 4. Convert image color from BGR to HSV

```

cvtColor(bottle_saved,      hsv_bottle_saved,      COLOR_BGR2HSV);
cvtColor(bottle1_saved,    hsv_bottle1_saved,    COLOR_BGR2HSV);
cvtColor(desabottle_saved,  hsv_desabottle_saved, COLOR_BGR2HSV);
cvtColor(desabottle1_saved, hsv_desabottle1_saved, COLOR_BGR2HSV);
cvtColor(desabottle2_saved, hsv_desabottle2_saved, COLOR_BGR2HSV);

cvtColor(can_saved,        hsv_can_saved,        COLOR_BGR2HSV);
cvtColor(can1_saved,      hsv_can1_saved,      COLOR_BGR2HSV);
cvtColor(can2_saved,      hsv_can2_saved,      COLOR_BGR2HSV);
cvtColor(can3_saved,      hsv_can3_saved,      COLOR_BGR2HSV);

cvtColor(can4_saved,      hsv_can4_saved,      COLOR_BGR2HSV);
cvtColor(can5_saved,      hsv_can5_saved,      COLOR_BGR2HSV);

cvtColor(glass_saved,     hsv_glass_saved,     COLOR_BGR2HSV);
cvtColor(glass1_saved,   hsv_glass1_saved,   COLOR_BGR2HSV);
cvtColor(glass2_saved,   hsv_glass2_saved,   COLOR_BGR2HSV);
cvtColor(glass3_saved,   hsv_glass3_saved,   COLOR_BGR2HSV);

```

## 5. Set the range of histogram size

```

int h_value =50;
int s_value =60;
int histsize[]={h_value, s_value};

float h_range[]={0,180};
float s_range[]={0,255};
const float* range[]={h_range, s_range};

int channel[]={0,1};

```

## 6. Assign name for histogram of database images

```
MatND hist_bottle_saved;
MatND hist_bottle1_saved;
MatND hist_desabottle_saved;
MatND hist_desabottle1_saved;
MatND hist_desabottle2_saved;
```

## 7. Calculate and normalize histogram of database images

```
calcHist(&hsv_bottle_saved, 1, channel, Mat(), hist_bottle_saved, 2, histsize, range, true, false);
normalize(hist_bottle_saved, hist_bottle_saved, 0, 1, NORM_MINMAX, -1, Mat());
calcHist(&hsv_bottle1_saved, 1, channel, Mat(), hist_bottle1_saved, 2, histsize, range, true, false);
normalize(hist_bottle1_saved, hist_bottle1_saved, 0, 1, NORM_MINMAX, -1, Mat());
```

## 8. Start-up camera and capture image, save computer address, assign name for captured image and histogram of captured images, convert captured image colour from BGR to HSV, calculate and normalize histogram of captured images.

```
cap >> frame;
//imshow("Recycle Object View", frame);
imwrite("C:/Users/Toshiba/Pictures/opencvtest1.jpg", frame);

Mat cam_input, hsv_cam_input;

cam_input = imread("C:/Users/Toshiba/Pictures/opencvtest1.jpg", CV_LOAD_IMAGE_COLOR);

cvtColor(cam_input, hsv_cam_input, COLOR_BGR2HSV);
MatND hist_cam_input;

calcHist(&hsv_cam_input, 1, channel, Mat(), hist_cam_input, 2, histsize, range, true, false);
normalize(hist_cam_input, hist_cam_input, 0, 1, NORM_MINMAX, -1, Mat());
```

## 9. Extract the comparison value between the histogram

```
double bottle_input = compareHist(hist_bottle_saved, hist_cam_input, 0);
double bottle1_input = compareHist(hist_bottle1_saved, hist_cam_input, 0);
double desabottle_input = compareHist(hist_desabottle_saved, hist_cam_input, 0);
double desabottle1_input = compareHist(hist_desabottle1_saved, hist_cam_input, 0);
double desabottle2_input = compareHist(hist_desabottle2_saved, hist_cam_input, 0);
double glass_input = compareHist(hist_glass_saved, hist_cam_input, 0);
double glass1_input = compareHist(hist_glass1_saved, hist_cam_input, 0);
double glass2_input = compareHist(hist_glass2_saved, hist_cam_input, 0);
double glass3_input = compareHist(hist_glass3_saved, hist_cam_input, 0);
```

10. Base on comparison value enter suitable "if loop" to send command to Arduino and save the data in notepad for further analysis.

```

if(bottle_input>=0.8 || bottle1_input>=0.8 || desabottle_input>=0.76 || desabottle1_input>=0.76 || desabottle2_input>=0.8)
{
    outputChars[0] = 'A';
    WriteFile(hSerial, outputChars, strlen(outputChars), &btsIOs, NULL);
    printf("It is a plastic Bottle\n");
    cout<<"Comparison (1) is " <<bottle_input<<endl;
    cout<<"Comparison (1.1) is " <<bottle1_input<<endl;
    cout<<"Comparison (2) is " <<desabottle_input<<endl;
    cout<<"Comparison (3) is " <<desabottle1_input<<endl;
    cout<<"Comparison (4) is " <<desabottle2_input<<endl;
    ofstream myfile;
    myfile.open ("TEXT.txt",ios::app);
    myfile << "It is a plastic bottle.\n";
    myfile << "Comparison (1) is " <<bottle_input<<endl;
    myfile << "Comparison (1.1) is " <<bottle1_input<<endl;
    myfile << "Comparison (2) is " <<desabottle_input<<endl;
    myfile << "Comparison (3) is " <<desabottle1_input<<endl;
    myfile << "Comparison (4) is " <<desabottle2_input<<endl<<endl;
    myfile.close();
    Sleep(20000);
    //system("pause");
}

```



Figure 4.1 describes the method to compare images from camera and databased. Due to limited space, the illustration below only shows four images from databased. In actual, there are 20 images (consists of Desa and 100plus plastic bottles, Milo and Pepsi aluminum cans and glass bottles) in databased need to be compared to decide the its categories. After comparison, output from window console will show the result which include the type my bottle and the comparison value. All these data will be transferred and saved in notepad. All these steps are repeated for next drinking bottle detection.

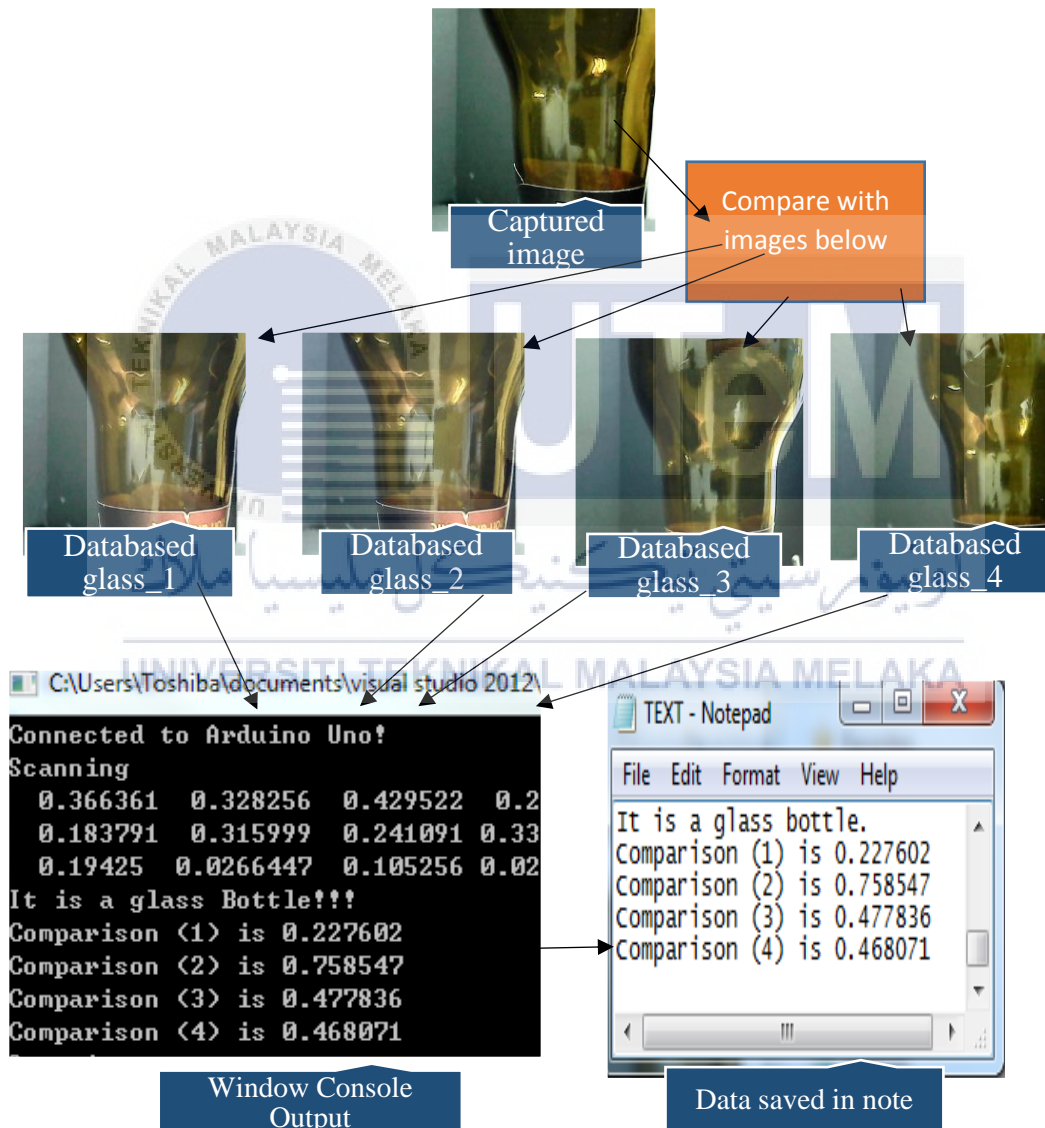


Figure 4.1: Method to compare images from camera and databased

## 4.1.1 Result from glass bottle test

Table 4.1 : Glass test 1 of ten input

Glass test (1)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)
Input 1	0.893386	0.829986	0.754037	0.791211
Input 2	0.844227	0.832412	0.91038	0.814125
Input 3	0.813642	0.745659	0.916526	0.814792
Input 4	0.889388	0.740083	0.798614	0.897275
Input 5	0.811782	0.659446	0.754968	0.896658
Input 6	0.55126	0.255411	0.479197	0.752004
Input 7	0.546657	0.248242	0.473032	0.751965
Input 8	0.554963	0.257085	0.484637	0.750342
Input 9	0.52157	0.271616	0.474142	0.760922
Input 10	0.883117	0.587478	0.748422	0.884918


Note: Minimum required value


Comparison (1)  $\geq 0.72$

Comparison (2)  $\geq 0.72$

Comparison (3)  $\geq 0.73$

Comparison (4)  $\geq 0.78$

 accepted data

 error detection


 comparison value didn't pass the min requirement



Table 4.2 : Glass test 2 of ten input

Glass test (2)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)
Input 1	0.498617	0.427182	0.796179	0.708744
Input 2	0.607095	0.535064	0.892886	0.702351
Input 3	0.553849	0.47546	0.862103	0.697976
Input 4	0.523665	0.527815	0.886723	0.666403
Input 5	0.54552	0.52439	0.909479	0.698991
Input 6	0.528435	0.558257	0.852071	0.617865
Input 7	0.641195	0.48888	0.756179	0.737843
Input 8	0.80773	0.561594	0.573471	0.631806
Input 9 (error)	-	-	-	-
Input 10	0.605533	0.371257	0.533791	0.833791


Note: Minimum required value


Comparison (1)  $\geq 0.72$

Comparison (2)  $\geq 0.72$

Comparison (3)  $\geq 0.73$

Comparison (4)  $\geq 0.78$

 accepted data

 error detection

 comparison value didn't pass the min requirement

Table 4.3 : Glass test 3 of ten input

Glass test (3)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)
Input 1	0.528073	0.325787	0.5879	0.840629
Input 2	0.623573	0.495911	0.651754	0.804042
Input 3	0.737909	0.558746	0.48394	0.574252
Input 4	0.769911	0.565767	0.500511	0.585278
Input 5	0.750374	0.533743	0.481476	0.608714
Input 6	0.787855	0.53926	0.467411	0.575985
Input 7 (error)	-	-	-	-
Input 8	0.750313	0.494382	0.479126	0.643115
Input 9	0.77183	0.54985	0.493626	0.626836
Input 10	0.295708	0.200931	0.732105	0.529733


Note: Minimum required value


Comparison (1)  $\geq 0.72$


Comparison (2)  $\geq 0.72$

Comparison (3)  $\geq 0.73$

Comparison (4)  $\geq 0.78$

 accepted data

 error detection

 comparison value didn't pass the min requirement

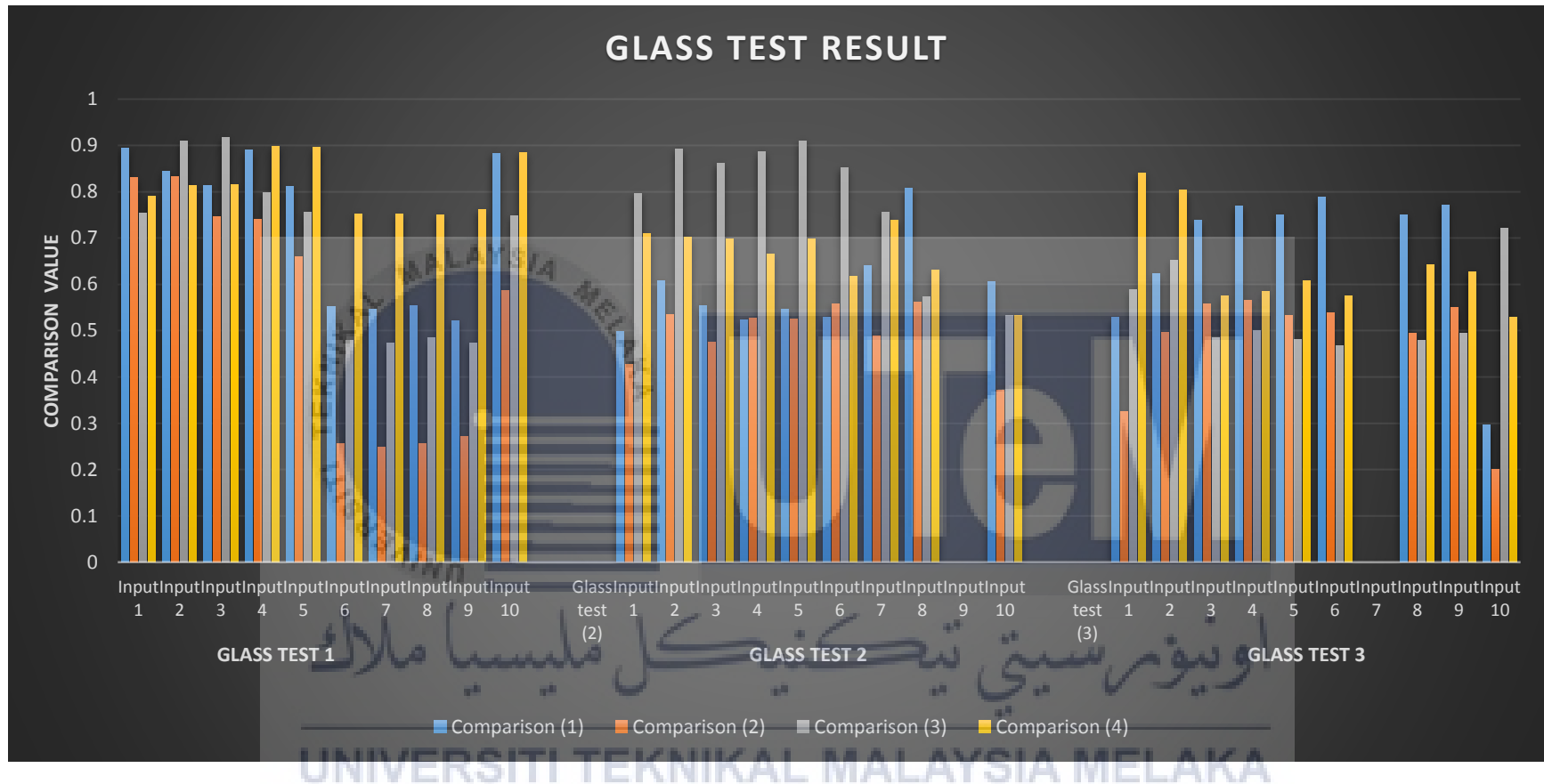


Figure 4.2: Histogram of glass test result of 30 input



Figure 4.3: Glass was ready to be transferred

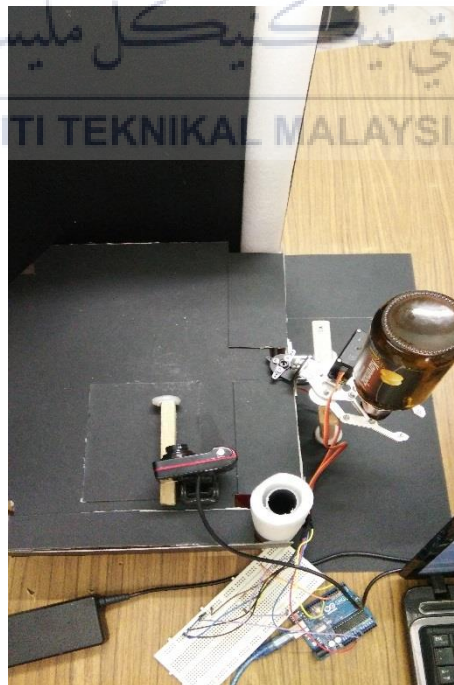


Figure 4.4: Glass rotated clockwise 225 degree

#### 4.1.2 Discussion from glass bottle test

From Table 4.1 (Glass test 1 of ten input), Table 4.2 (Glass test 2 of ten input) and Table 4.3 (Glass test 3 of ten input), the results show that all the testing able to achieve 90% and above success rate by detecting glass bottle. First testing achieves 100% rate of success while other two testing gain 90% success rate.

There are four different view of glass images in databased as shown in Appendix section 7.5. These images are compared with the captured imaged, which give four comparison value (Comparison 1, 2, 3, and 4). From the tables, light grey boxes indicate accepted data to prove the object detected in captured image is glass bottle. Dark grey boxes mean the data is undervalue which do not fulfil requirement. The reason to include four comparisons is due to different view of glass may have different result. If one of the comparison value passes minimum requirement, the system will record the data and send command to Arduino Uno through serial communication. Otherwise, system will recapture image and remake comparison.

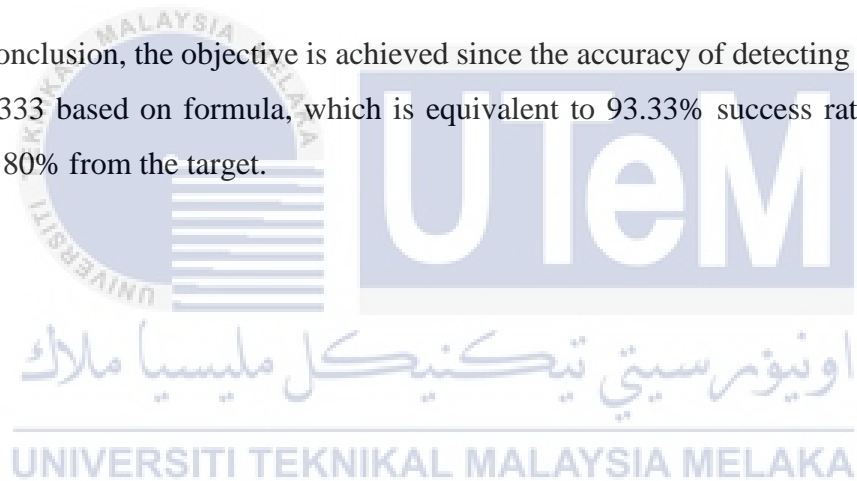
All the result from these tables are plotted into Figure 4.2 (Histogram of glass test result of 30 input) for future analysis. Based on this figure, Comparison (4) provides the most idea result of sensing glass bottle in first test while Comparison (2) give fluctuated result which is not stable. Looking at second test, Comparison (3) provides the highest comparison value while Comparison 2 is the lowest. Next, third test illustrated Comparison (1), (3) and (4) take turn to provide the ideal data to the system. From this experiment, it proves that the four images in databased are used as reference to recognise glass bottle due to it provides different view and scene of the same glass bottle.

The errors appeal in second and third test are due to image input does not match with image saved in database. Although the component is the same, but the colour of image captured is affected by light intensity and reflection of light. These three experiments are done in different room and different position. Hence, light intensity and reflection of light are the manipulated variable in this experiment.

Based on Figure 4.2, the minimum requirement of comparison value is set at 0.6 for glass bottle detection to reduce error detection. For improvement, the minimum value can be raised to 0.7 since only second test input number 10 does not pass this requirement. 0.7 is determined by doing 3 times of experiment with the theory behind of histogram comparison technique. In the command of “OpenCV compareHist”, the similarity of image colour between captured image and database images is calculated. The minimum comparison value (0.7) meaning that need at least 70% similarity of colour between captured image and databased image for the system to recognize the object.

By looking at Figure 4.3, gripper is gripping glass bottle and ready to transfer it while Figure 4.4 shows gripper successfully grip and rotated clockwise 225 degree to sort it. From these figures, system not only recognise it but also take proper action.

In conclusion, the objective is achieved since the accuracy of detecting glass bottle reaches 0.9333 based on formula, which is equivalent to 93.33% success rate, which is higher than 80% from the target.



## 4.1.3 Result from plastic bottle test


Table 4.4 : Plastic (DESA) test 1 of ten input

Plastic (Desa) (1)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)	Comparison (5)
Input 1	0.558759	0.720002	0.772666	0.432092	0.572062
Input 2	0.301681	0.667435	0.879419	0.153531	0.401531
Input 3	0.306382	0.665552	0.852103	0.151371	0.549871
Input 4	0.275909	0.662214	0.82701	0.208474	0.387674
Input 5	0.275859	0.607616	0.803361	0.163708	0.443708
Input 6	0.343528	0.635496	0.808575	0.211208	0.211208
Input 7	0.197819	0.587791	0.789785	0.517513	0.517513
Input 8	0.382175	0.636356	0.820206	0.548406	0.388534
Input 9	0.318545	0.612602	0.807136	0.548508	0.275834
Input 10	0.578134	0.754413	0.824976	0.586817	0.485217

Note: Minimum required value

Comparison (1)  $\geq 0.80$

Comparison (4)  $\geq 0.76$

 accepted data

Comparison (2)  $\geq 0.80$

Comparison (5)  $\geq 0.80$

 error detection

Comparison (3)  $\geq 0.76$

 comparison value didn't pass the min requirement

Table 4.5 : Plastic (100 PLUS) test 2 of ten input

Plastic (100plus) (2)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)	Comparison (5)
Input 1	0.848146	0.678721	0.233256	0.23542	0.271501
Input 2	0.778684	0.812141	0.345659	0.326125	0.351037
Input 3	0.826298	0.722654	0.328696	0.322945	0.327894
Input 4	0.769801	0.847763	0.389548	0.352041	0.354586
Input 5	0.883384	0.717446	0.356068	0.334131	0.369284
Input 6 (error)	-	-	-	-	-
Input 7	0.714977	0.812295	0.364943	0.351285	0.298638
Input 8	0.818584	0.782241	0.345659	0.326125	0.351037
Input 9	0.837156	0.658823	0.254256	0.25642	0.274301
Input 10	0.876298	0.767654	0.329496	0.325645	0.386894

Note: Minimum required value


Comparison (1)  $\geq 0.80$

Comparison (4)  $\geq 0.76$

 accepted data

Comparison (2)  $\geq 0.80$

Comparison (5)  $\geq 0.80$

 error detection

Comparison (3)  $\geq 0.76$


 comparison value didn't pass the min requirement




Table 4.6 : Plastic (DESA &amp; 100 PLUS) test 3 of ten input

Plastic (100plus & Desa) (3)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)	Comparison (5)
Input 1	0.826519	0.810622	0.343832	0.301157	0.702441
Input 2	0.353407	0.416607	0.472016	0.345351	0.809402
Input 3 error (can't detect 100plus)	-	-	-	-	-
Input 4	0.420186	0.51626	0.576578	0.438591	0.826712
Input 5	0.732263	0.87408	0.46087	0.409061	0.684192
Input 6	0.458753	0.546629	0.541617	0.44874	0.877623
Input 7	0.435239	0.612818	0.651783	0.559221	0.846746
Input 8	0.81318	0.7157	0.261851	0.258587	0.705286
Input 9	0.437555	0.60732	0.641001	0.554605	0.826001
Input 10	0.706026	0.893662	0.435192	0.410617	0.708521

Note: Minimum required value

Comparison (1)  $\geq 0.80$

Comparison (4)  $\geq 0.76$

 accepted data

Comparison (2)  $\geq 0.80$

Comparison (5)  $\geq 0.80$

 error detection

Comparison (3)  $\geq 0.76$

 comparison value didn't pass the min requirement

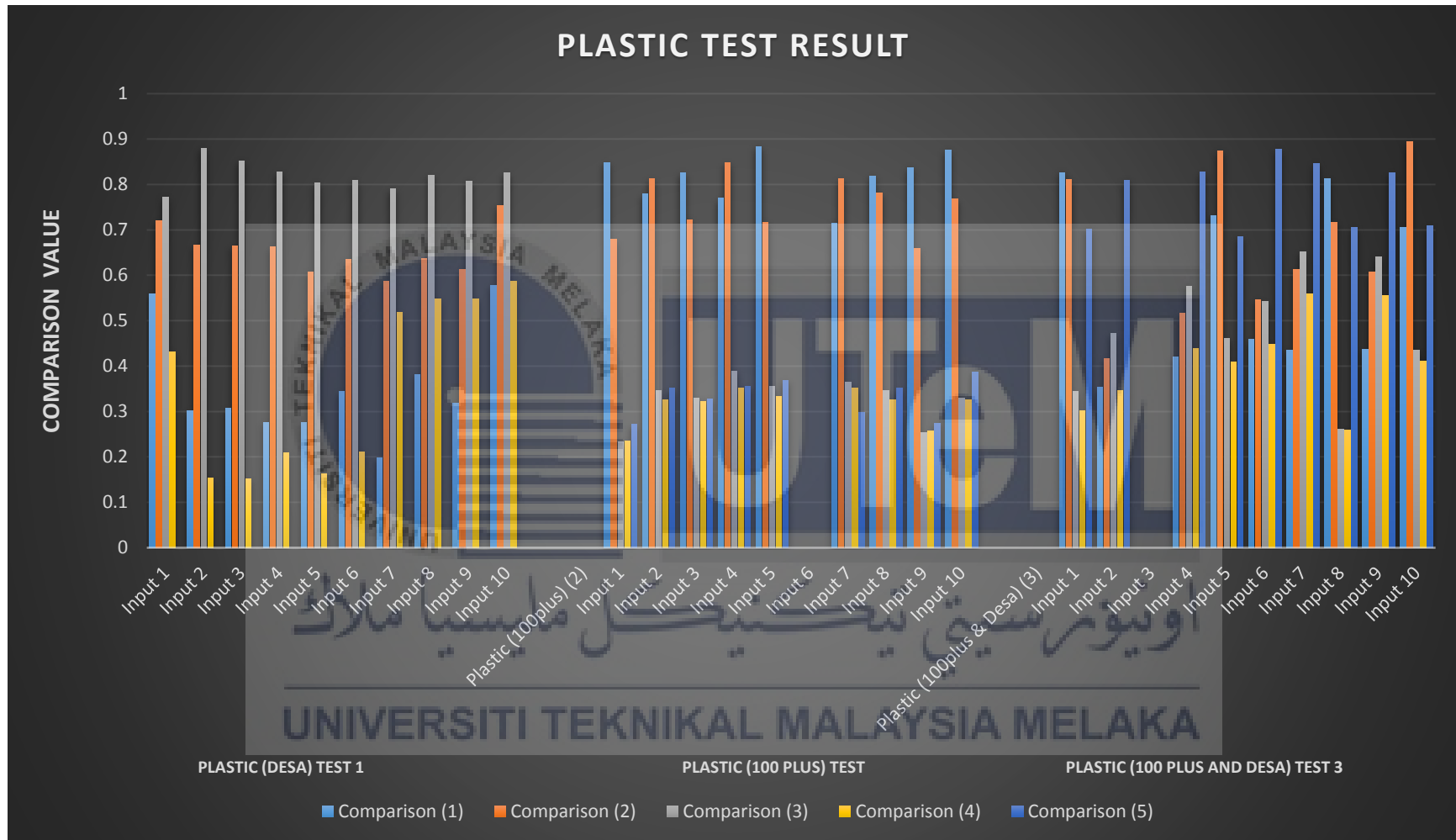


Figure 4.5: Histogram of 30 input plastic test result

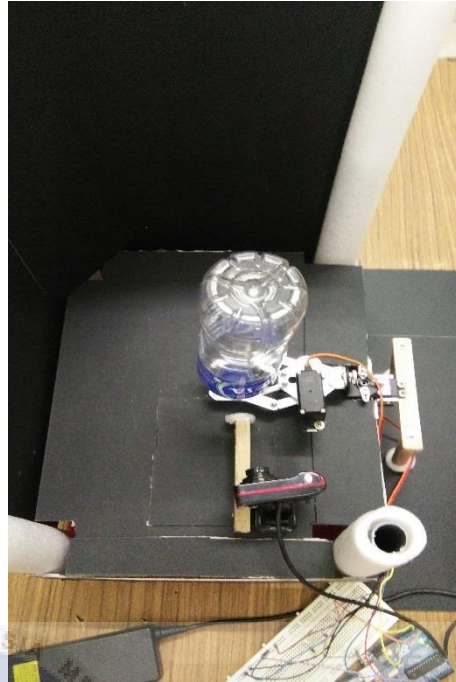


Figure 4.6: Plastic bottle was ready to be transferred

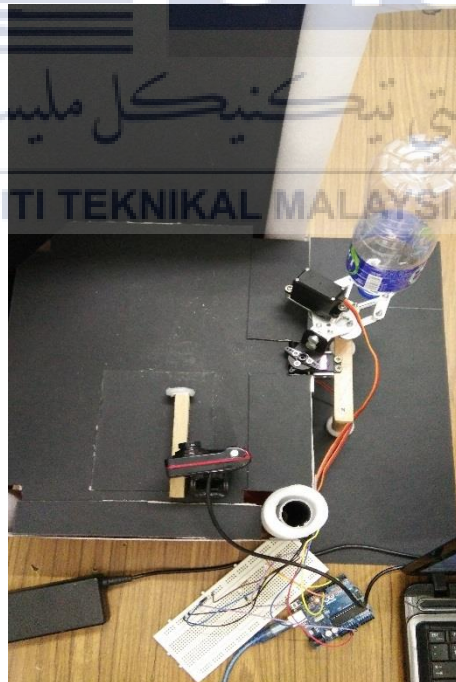


Figure 4.7: Plastic bottle rotated clockwise 135 degree

#### 4.1.4 Discussion from plastic bottle test

From Table 4.4 record the ten inputs result for plastic bottle (brand DESA), Table 4.5 record the ten inputs result for plastic bottle (brand 100 plus) and Table 4.6 record the ten inputs result for plastic bottle (brand DESA and 100 plus), the data illustrate that all the testing able to achieve 90% and above success rate by detecting plastic bottle. First testing of detecting DESA bottles reached 100% rate of success while the rest gain 90% success rate.

In the case of detecting plastic bottle, there are five different view of plastic bottle images in databased, two from 100plus bottles and three from Desa bottles as shown in Appendix section 7.5. These images are compared with the captured imaged, which give five comparison value (Comparison 1, 2, 3, 4, and 5). From the tables, light grey boxes indicate accepted data to prove the object detected is plastic bottle. Dark grey boxes mean the data is undervalue which do not fulfil requirement.

Light grey box appeals in Comparison (1) or (2) meaning that it is 100plus plastic bottle, else, it is Desa plastic bottle. The purpose to include five comparisons is to speed up the detection and increase accuracy. Although the images in databased may look like similar, but different view and scene of bottle may affect overall performance. If one of the comparison value passes minimum requirement, the system will record the data and send command to Arduino Uno through serial communication. Otherwise, system will recapture image and remake comparison.

The three tables are plotted into Figure 4.5 (Histogram of plastic test result of 30 input) for future analysis. By looking at the figure, Comparison (3) provided the most idea result of sensing plastic bottle in first test while Comparison (1) and (4) gave fluctuated result. Moving to second test, both comparison value from (1) and (2) higher than 0.6 while other fail to reach 0.4. Next, third test illustrates Comparison (1), (2) and (5) took turn to provide the ideal data to the system. From this experiment, it shows that the Comparison (4) do not provide positive result throughout three experiments. It can be replaced by another view of Desa bottle images in databased to rise accuracy and speed of detection.

The errors occur in second and third test are due to system cannot recognize input image. Although the component (100plus) in the image captured is the same, but the colour of image is affected by light intensity and reflection of light, same as the error occur in pervious test. These three experiments are done in different environment and different position. Therefore, light intensity and reflection of light are the factors to affect sensitivity of detection.

In Figure 4.5, the minimum requirement of comparison value can be reduced and set at 0.75 for plastic bottle detection to reduce error detection. Its value is slightly higher than value of glass bottle detection due to low reflection index. Plastic detection more reliable and stable. 0.75 is determined by doing 3 times of experiment with the theory behind histogram comparison technique. In the command of “OpenCV compareHist”, the similarity of image colour between captured image and database images is calculated. Value equals to 1 shows that comparison are totally same. The minimum requirement of comparison value 0.75 meaning that need at least 75% similarity of colour between captured image and databased image for the system to recognize the object.

By looking at Figure 4.6, gripper is gripping plastic bottle and ready to transfer it while Figure 4.7 illustrate gripper successfully grip and rotate clockwise 135 degree to sort it. From these figures, system not only recognise it but also take right action to sort it.

Once again, the objective is achieved for plastic bottle detection as the result reaches 93.33% success rate, which is higher than 80% from the target.

## 4.1.5 Result from aluminium can test


Table 4.7: Can (Pepsi) test 1 of ten input

Can (Pepsi) (1)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)	Comparison (5)	Comparison (6)
Input 1	0.129681	0.0716599	0.243354	0.229671	0.775401	0.667384
Input 2	0.217882	0.181096	0.448894	0.411778	0.923185	0.583209
Input 3	0.193653	0.142675	0.398719	0.365321	0.89547	0.530819
Input 4	0.126663	0.114267	0.337318	0.315933	0.784506	0.310834
Input 5	0.157462	0.161386	0.406727	0.399892	0.929267	0.464802
Input 6	0.143876	0.0556567	0.238513	0.20061	0.713549	0.714252
Input 7	0.134153	0.0611262	0.257724	0.223955	0.718847	0.501287
Input 8	0.0768783	0.0516099	0.200538	0.188163	0.71018	0.595252
Input 9	0.157685	0.105435	0.298951	0.277046	0.777313	0.687966
Input 10	0.108563	0.0366135	0.198901	0.158476	0.682336	0.696729

Note: Minimum required value

Comparison (1)  $\geq 0.68$

Comparison (4)  $\geq 0.68$

 accepted data

Comparison (2)  $\geq 0.68$

Comparison (5)  $\geq 0.68$

 error detection

Comparison (3)  $\geq 0.68$

Comparison (6)  $\geq 0.68$


 comparison value didn't pass the min requirement

Table 4.8: Can (Milo) test 2 of ten input

Can (Milo) (2)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)	Comparison (5)	Comparison (6)
Input 1	0.48172	0.543393	0.725363	0.576462	0.492731	0.101609
Input 2	0.565435	0.550561	0.799938	0.449119	0.276177	0.0852412
Input 3	0.695819	0.709557	0.787755	0.525638	0.227934	0.0726457
Input 4	0.611646	0.58937	0.83153	0.627686	0.514934	0.147652
Input 5	0.55159	0.556686	0.880467	0.690706	0.51746	0.138954
Input 6	0.466599	0.559464	0.781995	0.710453	0.501683	0.108418
Input 7	0.397664	0.377502	0.600344	0.534687	0.715181	0.174993
Input 8	0.246293	0.292943	0.471559	0.785897	0.782059	0.151128
Input 9	0.390152	0.484151	0.697514	0.77992	0.669501	0.178308
Input 10	0.572455	0.650143	0.807201	0.340206	0.157666	0.0541806

Note: Minimum required value

Comparison (1)  $\geq 0.68$

Comparison (4)  $\geq 0.68$

 accepted data

Comparison (2)  $\geq 0.68$

Comparison (5)  $\geq 0.68$

 error detection

Comparison (3)  $\geq 0.68$

Comparison (6)  $\geq 0.68$


 comparison value didn't pass the min requirement


Table 4.9: Can (Milo &amp; Pepsi) test 3 of ten input

Can (Milo & Pepsi) (3)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)	Comparison (5)	Comparison (6)
Input 1	0.160428	0.0384804	0.129668	0.928219	0.178448	0.697493
Input 2	0.222563	0.0257374	0.17257	0.665355	0.329533	0.97523
Input 3	0.695734	0.750937	0.739828	0.099734	0.110491	0.156662
Input 4	0.853691	0.570602	0.704244	0.140861	0.224547	0.314055
Input 5	0.198958	0.0222906	0.158274	0.123078	0.718769	0.329344
Input 6	0.164123	0.0321586	0.15125	0.130063	0.938741	0.328343
Input 7	0.707442	0.492262	0.680623	0.166762	0.236214	0.291498
Input 8	0.663981	0.697587	0.56223	0.100268	0.168127	0.180973
Input 9	0.16438	0.0452811	0.127578	0.230485	0.753446	0.398406
Input 10	0.935427	0.666632	0.676519	0.114035	0.153981	0.250713

Note: Minimum required value

Comparison (1)  $\geq 0.68$

Comparison (4)  $\geq 0.68$

 accepted data


Comparison (2)  $\geq 0.68$

Comparison (5)  $\geq 0.68$

 error detection

Comparison (3)  $\geq 0.68$

Comparison (6)  $\geq 0.68$

 comparison value didn't pass the min requirement



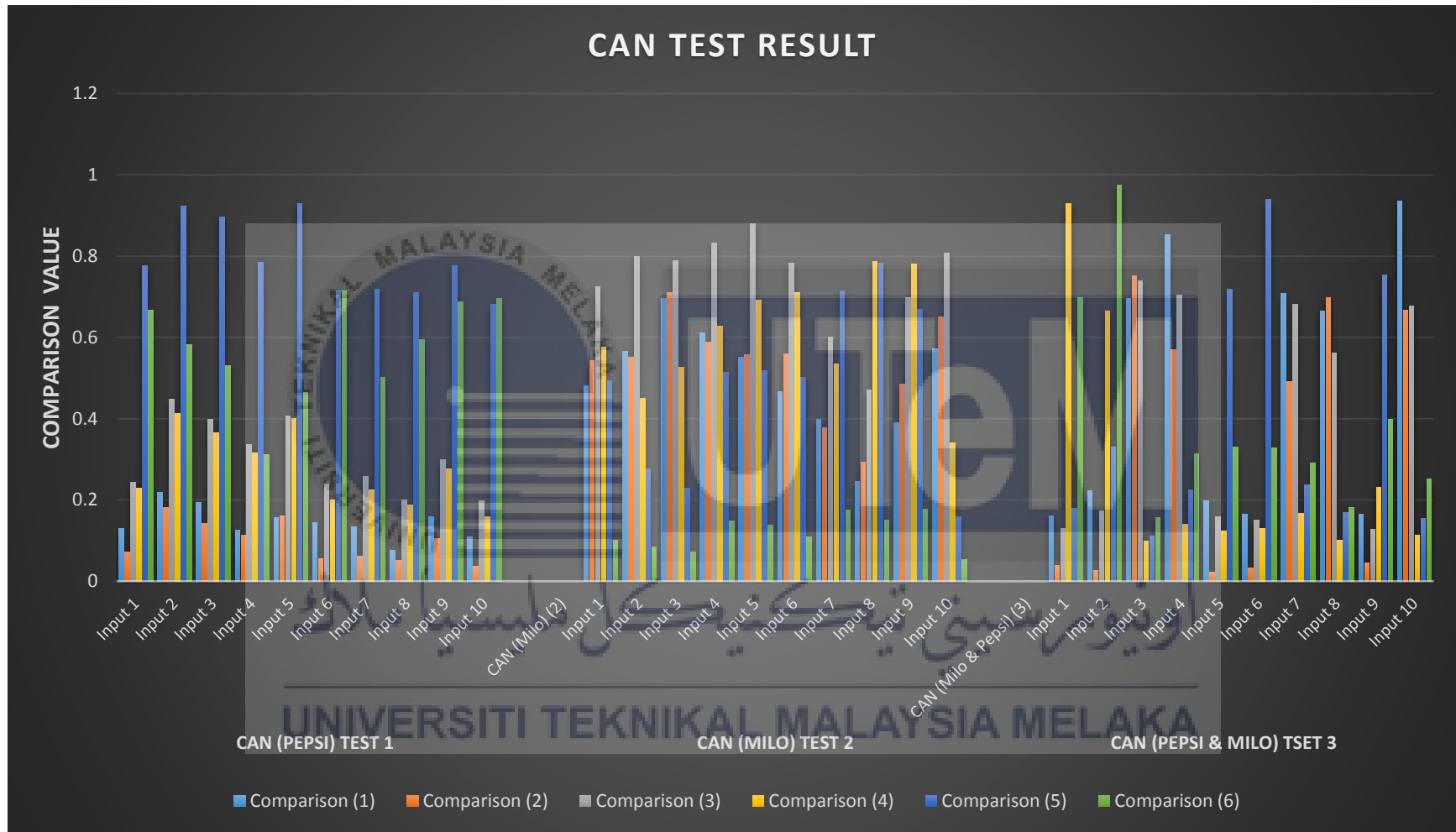


Figure 4.8: Histogram of 30 input can test result



Figure 4.9: Aluminium tin was ready to be transferred

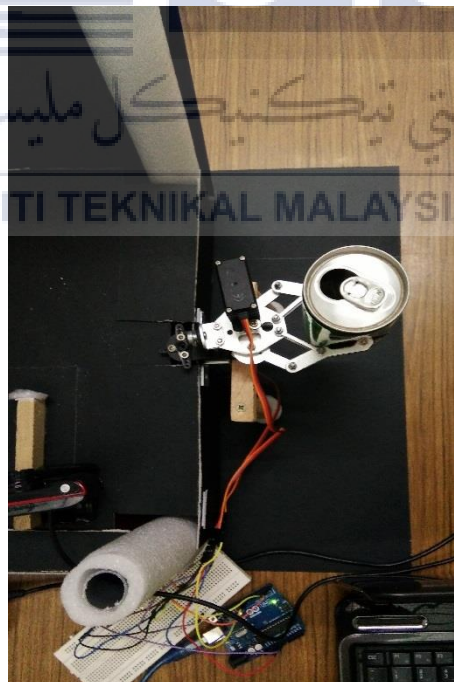


Figure 4.10: Aluminium tin rotated clockwise 180 degree

#### 4.1.6 Discussion from aluminium can test

Referring to Table 4.7 above which records the ten inputs result for aluminium can (brand Pepsi), Table 4.8 records the ten inputs result for aluminium can (brand Milo) and Table 4.9 records the ten-input result for aluminium can (brand Pepsi and Milo), the data illustrates that all the testing achieve 100% success rate to detect the input cans.

Considering detection of aluminium can, there are six different view of aluminium can images in databased, three from Milo and three from Pepsi as shows in Appendix section 7.5. These images are compared with the captured imaged, which give six comparison value (Comparison 1, 2, 3, 4, 5 and 6). From the tables, light grey boxes indicate accepted data to prove the object detected is aluminium. Dark grey boxes mean the data is undervalue which do not fulfil requirement. Light grey boxes appeal in Comparison (1), (2) and (3) meaning that it is Milo can, else, it is Pepsi can. Many comparisons are added to let the system goes smooth and reduce error detection. Nevertheless, the images in databased may look similar, but the view may affect overall performance. The system just need either one of the comparison value passes minimum requirement, the system will record the data and send command to Arduino Uno through serial communication. Else, it will repeat the scanning process.

Figure 4.8 (Histogram of plastic test result of 30 input) is plotted based on the result of Tables 4.7, Table 4.8 and Table 4.9. The figure shows that Comparison (5) is the most suitable image to detect Pepsi can while Comparison (2) is the worst. Moving to second test, Comparison (3) and (4) are more suitable to detect Milo cans as other Comparison shown fluctuated result. Next, third test illustrated all types of Comparison are needed to detect randomly input of two different aluminium can. This testing proves that the six images in databased are fully used as reference to recognise aluminium can due to it provides different view and angle of the same aluminium can.

There is no error occurred in these tests. This is probably due to the components (Pepsi and Milo cans) have contrast colour and shining surface. Thus, the input image and saved image in database has higher similarity. Although these three experiments were done

in different environment and different position, but light intensity and reflection of light do not affect the experiment so much for detecting aluminium can.

In Figure 4.8, the minimum requirement of comparison value can be set at 0.62 for aluminium can detection to reduce error detection. 0.62 is determined by running experiment with the theory behind of histogram comparison technique. In the command of “OpenCV compareHist”, the similarity of image colour between captured image and database images is calculated, which means that the more similar between captured image and databased image, the higher the value of comparison.

0.62 value stand between the value of glass bottle detection and plastic bottle detection. Although it does not have very high comparison value to show higher similarity, but this value is reliable to let the system know the detected component is aluminium can. By looking at Figure 4.9, gripper is gripping aluminium can and ready to transfer it while Figure 4.10 illustrate gripper successfully grip and rotate clockwise 180 degree to sort it. From these figures, system able to recognise it but also take right action to sort it.

Once again, the objective is achieved for aluminium can detection as the result reaches 100% success rate, which is higher than the expected result.

## 4.1.7 Result from overall accuracy testing

Table 4.10: Final (mixed bottles) test 1 of ten input

Final (1)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)	Comparison (5)	Comparison (6)
Input 1 (Glass)	0.246736	0.285983	0.750255	0.507446		
Input 2 (Plastic)	0.782114	0.85074	0.461188	0.456376	0.661311	
Input 3 (Plastic)	0.829093	0.647921	0.417168	0.361928	0.624787	
Input 4 (Can)	0.21708	0.0720239	0.16021	0.396895	0.896372	0.620115
Input 5 (Can)	0.247922	0.0685073	0.191286	0.34811	0.885143	0.594526
Input 6 (Plastic)	0.852288	0.94855	0.371517	0.397693	0.726166	
Input 7 (Glass)	0.927145	0.698043	0.489522	0.38348		
Input 8 (Plastic)	0.560765	0.801267	0.123404	0.233907	0.49566	
Input 9 (Can)	0.725293	0.36069	0.580379	0.135119	0.320704	0.407857
Input 10 (Glass)	0.80892	0.790892	0.392154	0.297737		

Note:

- UNIVERSITI TEKNIKAL MALAYSIA MELAKA
- Glass testing (comparison value  $\geq 0.6$ )
  - Plastic testing (comparison value  $\geq 0.75$ )
  - Aluminum testing (comparison value  $\geq 0.62$ )
  - comparison value didn't pass the min requirement
  - not related
  - accepted data
  - error

Table 4.11: Final (mixed bottles) test 2 of ten input

Final (2)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)	Comparison (5)	Comparison (6)
Input 1 (Glass)	0.261575	0.295651	0.816623	0.553193		
Input 2 (Glass)	0.276074	0.296314	0.828814	0.539676		
Input 3 (Plastic)	0.697662	0.938378	0.2942	0.322359	0.60873	
Input 4 (Plastic)	0.849457	0.724718	0.424945	0.382301	0.669968	
Input 5 (error)	-	-	-	-	-	-
Input 6 (Can)	0.354981	0.125661	0.284106	0.312978	0.680791	0.674487
Input 7 (Can)	0.917812	0.520446	0.666748	0.151645	0.295409	0.410289
Input 8 (Can)	0.288132	0.0789969	0.201378	0.201378	0.706385	0.55931
Input 9 (Can)	0.24708	0.063443	0.18909	0.401875	0.776873	0.624856
Input 10 (Can)	0.149662	0.032432	0.119113	0.855148	0.566622	0.62127

Note:

- |   |  |   |  |
|---|--|---|--|
|  | Glass testing (comparison value $\geq 0.6$ )     |  | comparison value didn't pass the min requirement |
|  | Plastic testing (comparison value $\geq 0.75$ )  |  | not related                                      |
|  | Aluminum testing (comparison value $\geq 0.62$ ) |  | accepted data                                    |
|   |  |  | error  |

Table 4.12: Final (mixed bottles) test 3 of ten input

Final (3)	Comparison (1)	Comparison (2)	Comparison (3)	Comparison (4)	Comparison (5)	Comparison (6)
Input 1 (Plastic)	0.584995	0.806814	0.180515	0.252089	0.482847	
Input 2 (Can)	0.850059	0.637955	0.58567	0.0773033	0.191435	0.249629
Input 3 (Plastic)	0.676332	0.917567	0.237294	0.251196	0.627659	
Input 4 (Glass)	0.882779	0.691179	0.321044	0.246414		
Input 5 (Plastic)	0.821125	0.935696	0.256709	0.278214	0.6635	
Input 6 (Can)	0.804396	0.489073	0.57815	0.102907	0.245954	0.332646
input 7 (error)	-	-	-	-	-	-
Input 8 (Can)	0.29391	0.112543	0.225177	0.268348	0.737576	0.538309
Input 9 (Can)	0.42167	0.124553	0.328811	0.294947	0.71311	0.577528
Input 10 (Glass)	0.790824	0.72365	0.364771	0.299283		

Note:

- Glass testing (comparison value  $\geq 0.6$ )
  comparison value didn't pass the min requirement
- Plastic testing (comparison value  $\geq 0.75$ )
  not related
- Aluminum testing (comparison value  $\geq 0.62$ )
  accepted data
- error

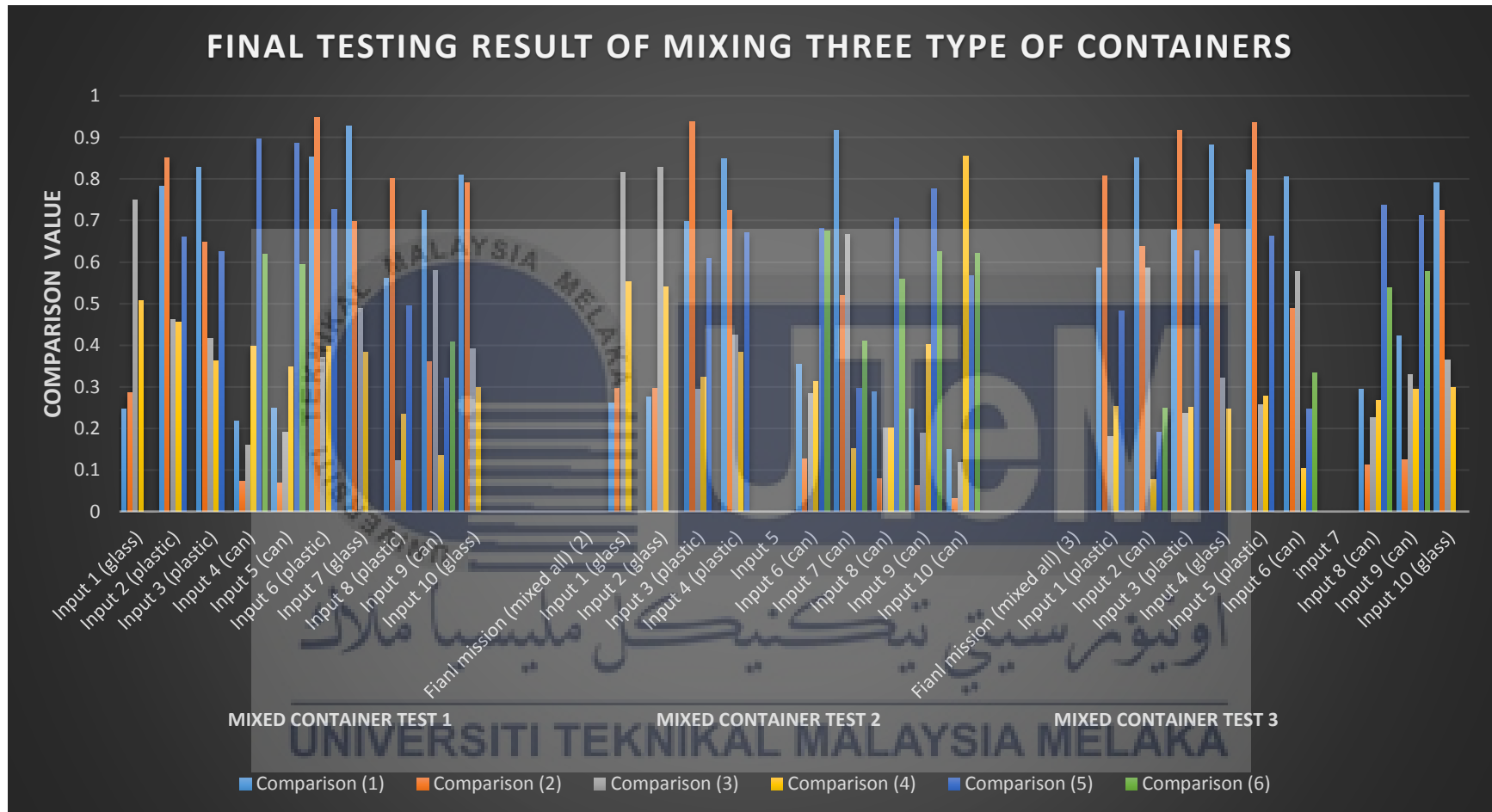


Figure 4.11: Histogram of 30 input mixed bottles test result



#### 4.1.8 Discussion from final test (mixed bottle)

Referring to Table 4.10, Table 4.11 and Table 4.12 above, which recorded the ten inputs result for mixed bottles respectively. The data illustrated first testing achieve 100% success rate while the second and third test reached 90% of success rate.

Figure 4.11 (Histogram of 30 input mixed bottles test result) is plotted based on the data from Tables 4.10, Table 4.11 and Table 4.12. In first test, this system able to randomly detect four plastic bottles, three cans and three glass bottles with no delay and error. But the figure shown that the system occurs an error in second and third test respectively. It cannot recognize the type container.

As mention in previous section, error occur is due to influence of light intensity and reflection of light. These three experiments were also done in different environment and different position. Therefore, light intensity and reflection of light are the factors to affect sensitivity of detection. The main purpose of Figure 4.11 is to illustrate the sensitivities and reliabilities of this system. Comparison value to recognize glass, plastic and aluminium bottles or cans are fixed in pervious testing. Therefore, this histogram proved this design can accurately differentiate these drinking containers correctly.

Histogram is important to represent tonal distribution of digital image. It consists information of pixels' number for each tonal value. Horizontal axis of histogram graph represents colour (tonal) variation while number of pixel value is shown in vertical axis. Tonal is defined as variation of colour. Hence, histogram comparison technique used in these 12 experiments is about to compare the similarity of image colour between captured image and database images. The more similar between captured image and databased image, the higher the value of comparison calculated by "OpenCV compareHist" instruction. By knowing the theory behind, minimum comparison value can be easily set as done in 12 experiments previously to increase the accuracy of detection.

Objective is achieved for drinking containers detection as the result reached 93.33% success rate, which is higher than the target result. All these experiments reflect this project is reliable to use in the future as it only gives minor error by testing 120 times.

## CHAPTER 5

### CONCLUSION AND RECOMMENDATION

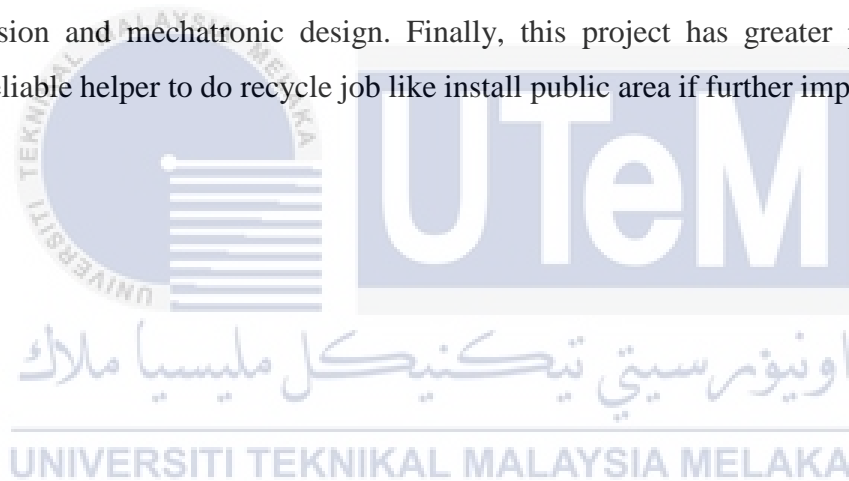
#### 5.1 Conclusion

After studied basic knowledge in machine vision and mechatronic subject, Automated Waste Separated Machine for final year project is successfully developed. This mechanism able to recognise object based on machine vision approach which achieved the first objective. Moreover, it favourably separate unbent or slightly bent waste plastic bottles, glass bottles, and aluminium cans by using machine vision which gained the second objective. Last objective also achieved as the system performance is well analysed. Result shown in Chapter 4 revealed that it having the quality to sort the target drinking bottle into category based on their type of material. Result illustrated that the minimum success rate is 0.9933 while the highest success rate up to 1. These data prove that machine vision having the potential to replace human to do that recycle job, especially in sorting waste drinking bottle.

However, this design is a fundamental mechanism only emphasize on certain volume and brand of drinking containers, which only included 500ml DESA and 100Plus plastic bottles, 325ml dark brown colour glass bottles, and 325ml Pepsi and Milo aluminium cans. But this is not the limitation of this mechanism design, new target drinking can or bottle image can save in database as reference for detection. For example, if the system want to sort new target like Coca-Cola can into its category, take one image of it and save in database. After that run a testing to test reliability to detect Coca-Cola can. Make troubleshooting if necessary.

There are a lot of techniques in writing image processing program for object detection. In this project design, the easiest method is chosen. Covert colour of image into HSV and then transform into histogram for comparison is the one used in this project. Although the program write in C++ language using OpenCV library seem longer, but it easier to be understood and encounter problem compare using Deep Learning method. Furthermore, output command can be easily added and modified to desired section in the program. Command will be sent through serial communication to Arduino controller to grip and move drinking bottle or can to target area.

In conclusion, all the objectives are achieved as the design can process image, recognize object and sort drinking container by itself. After finished making this prototype of Automated Waste Separated Machine, a lot good experience is gained, especially in machine vision and mechatronic design. Finally, this project has greater potential to become a reliable helper to do recycle job like install public area if further improvement is considered.



## 5.2 Recommendation

There are few part can be improved in the future. First of all, original gripper can be replaced by bigger one to hold larger size of bottle. Since in Malaysia online market like Lelong.com, the biggest size of gripper is same as the one used in this project. Thus, bigger gripper can be designed and drawn by Solidwork. After that use the drawing to make it by 3-D printer. Custom design gripper able to pick and place larger size of bottle like 1500ml of plastic drinking bottle which overcome the limitation of detecting small drinking container in this project.

Next, a fully cover black box with internal building lighting can be built to replace half cover original design base. As discussed in Chapter 4 result, light intensity and reflection of light are the major concern in image processing. From the experiment, the fluctuated comparison values are due to the experiment done in different environment and position. Light intensity and reflection of light become the manipulated variables that affecting accuracy of the mechanism. Hence, fully cover black box able to turn manipulated variable to fixed variable.

Furthermore, another C++ program method like Deep Learning can be written and analysed. Analyse the performance between Deep Learning method and the program that written in this project by using the same platform.

Last but not lease, machine vision is one of a good tool to detect object. Adding extra device like inductive proximity sensor to detect aluminium is an advantage to increase the sensitivities of the overall performance.

## REFERENCES

- [1] L. Spilsbury, Waste and recycle challenges, New York: The Rosen Publishing Group Inc. , 2010.
- [2] Qunzhou YU, Haibin CHEN, “Problem on Municipal Solid Waste Landfill,” Analysis on the Old Simple Municipal Solid Waste Landfill, p. 32, 2009.
- [3] “National Environment Agency (NEA),” 08 September 2016. [Online]. Available: <http://www.nea.gov.sg/energy-waste/waste-management/overview>.
- [4] I. ISMAIL, “Malaysians producing more solid waste than before,” Malay Mail Online , KUALA LUMPUR, 2012.
- [5] “Melaka historic city council,” 14 Dec 2016. [Online]. Available: <http://www.mbmb.gov.my/en/citizens/services/waste-management>.
- [6] Wei Zhao, Qun Zhang, “Solid Waste Management,” Solid Waste Management in Tianjin City , p. 3, 2011.
- [7] Min LOU and Qiao-Lun GU, “Investigation and analysis of university students’ cognition and behavior of the classification and recycling of solid waste,” 8th International Conference on Intelligent Networks and Intelligent Systems, pp. 165-168, 2015 .
- [8] W. H. Organization, “Fundamentals of health-care waste management,” United Nations Environment Programme / SBC, p. 8, 1999.
- [9] S. Boh, “Food waste raises a stink for recycling,” The Straits times , Tuas, 2016.
- [10] JMBarcala, JLFern´andez, JAlberdi, J Jim´enez, JCL´azaro, J J Navarrete and J C Oller, “Identification of plastics using wavelets,” MEASUREMENT SCIENCE AND TECHNOLOGY, p. 371–376, 2004.
- [11] Y. Tachwali, Y. Al-Assaf and A.R. Al-Ali, “Automatic multistage classification system,” Resources, Conservation and Recycling, pp. 266-285, 2007.
- [12] Matti Kutila, Jouko Viitanen and Antero Vattulainen, “Scrap Metal Sorting with Colour Vision and Inductive Sensor Array,” CIMCA-IAWTIC, 2005.
- [13] W. Bolton, Mechatronic electronic control systems in mechanical and electrical engineering, 4th ed., 2008.

- [14] E. Nayagam, "Quora," 15 March 2015. [Online]. Available: <https://www.quora.com/What-are-the-main-differences-between-hydraulic-and-pneumatic-Why-are-hydraulics-more-widely-used>.
- [15] Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing, United State of America: Prentice-Hall, Inc., 2002.
- [16] S. Ramli, Mohd Marzuki Mustafa, Aini Hussain and Dzuraidah Abdul Wahab, "Histogram of Intensity Feature Extraction for Automatic Plastic Bottle," American Journal of Environmental Sciences, pp. 583-588, 2008.
- [17] Rafael C. Gonzalez, Richard E. Wood, Digital Image Processing, United States of America: Prentic-Hall,Inc, 2002.
- [18] Edgar Scavino, Dzuraidah Abdul Wahab, Hassan Basri, Mohd Marzuki Mustafa and Aini Hussain, "A Genetic Algorithm for the Segmentation of Known Touching Objects," Journal of Computer Science, pp. 711-716, 2009.
- [19] Masami Sakurai, Yukio Kurihara and Shiro Karasawa, "Color Classification Using Fuzzy Inference and Genetic Algorithm," Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on, Orlando, FL, vol. 3, pp. 1975-1978, 1994.
- [20] Mohd Asyraf Zulkifley, Mohd. Marzuki Mustafa and Aini Hussain, "Probabilistic White Strip Approach to Plastic Bottle Sorting System," IEEE, pp. 3162-3164, 2013.
- [21] Hoo Seng Choon, Mohd Firduas Zakaria and Shahrel Azmin Suandi, "Object Shape Recognition in Image for Mahine Vision Application," International Journal of Computer Theory and Engineering, vol. 4, pp. 76-80, 2012.
- [22] "Arduino + Servo + openCV Tutorial [openFrameworks]," 28 Jun 2010. [Online]. Available: <http://www.creativeapplications.net/tutorials/arduino-servo-opencv-tutorial-openframeworks/>.
- [23] "Face detection and tracking with Arduino and OpenCV," 20 February 2013. [Online]. Available: <http://www.instructables.com/id/Face-detection-and-tracking-with-Arduino-and-OpenC/>.
- [24] "ipmart.com.my," [Online]. Available: [http://www.ipmart.com.my/main/product/Logitech\\_C920\\_HD\\_Pro\\_Webcam\\_Logitech\\_Warranty\\_351826.php?prod=351826&gclid=CjwKEAiAg5\\_CBRDo4o6e4o3NtG0SJAB-IatYouVkSiYSTGhP0GACZ-oA1kQS3fWqKZYTj4PWk6JEvhoCcUHw\\_wcB](http://www.ipmart.com.my/main/product/Logitech_C920_HD_Pro_Webcam_Logitech_Warranty_351826.php?prod=351826&gclid=CjwKEAiAg5_CBRDo4o6e4o3NtG0SJAB-IatYouVkSiYSTGhP0GACZ-oA1kQS3fWqKZYTj4PWk6JEvhoCcUHw_wcB).

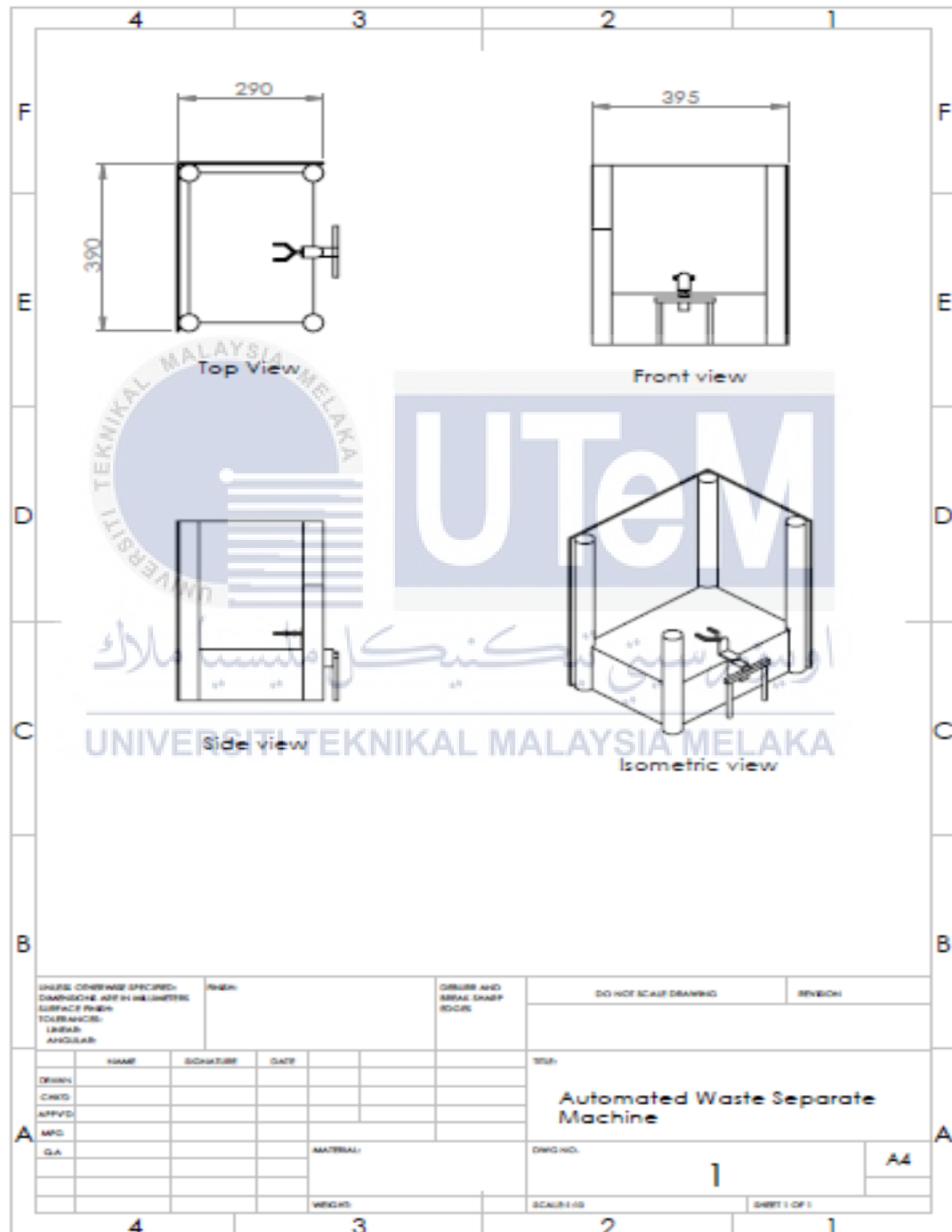
- [25] V. d. team, "Introduction: What is VXL?," 30 September 2013. [Online]. Available: <http://vxl.sourceforge.net/>.
- [26] L. d. team, "Introduction, 2013," 04 May 2010. [Online]. Available: <http://ltilib.sourceforge.net/doc/homepage/index.shtml>.
- [27] O. D. Team, "OpenCV," 10 September 2016. [Online]. Available: <http://opencv.org/about.html>.
- [28] S. Fernando, "OpenCV Tutorial C++," 15 April 2013. [Online]. Available: <http://opencv-srf.blogspot.my/p/opencv-c-tutorials.html>.
- [29] A. RAIZEN, "DiscoverSDK.com," DiscoverSDK Tools for Developer, 2016. [Online]. Available: <http://www.discoversdk.com/compare/opencv-vs-vxl>.
- [30] M. J. Franchetti, A system approach solid waste, United State : The McGraw-Hill Companies, Inc, 2009.
- [31] S. Shahbudin, A. Hussain, D. A. Wahab, M. M. and S. Ramli, "Support vector machines for," 6th International Colloquium on Signal Processing and Its Applications, pp. 1-5, 2010.
- [32] B.W. House, D.W. Capson and D.C. Schuurman, "Towards real-time sorting of recyclable goods using support vector machines," Sustainable Systems and Technology, pp. 1-6, 2011.

اوتنور سیتی تکنیکل ملیسیا ملاک

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

APPENDIX

7.1 Solidwork drawing of product outlook





## 7.2 Coding for Automated Waste Separated Machine in C++

```

#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <Windows.h>
#include <fstream>

using namespace std;
using namespace cv;

int main( int argc, char** argv )
{
    HANDLE hSerial = CreateFile(L"COM8", GENERIC_READ |
GENERIC_WRITE, 0, 0, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, 0);
    if (hSerial !=INVALID_HANDLE_VALUE)
    {
        printf("Connected to Arduino Uno! \n");

        DCB dcbSerialParams;
        GetCommState(hSerial,&dcbSerialParams);

        dcbSerialParams.BaudRate = CBR_9600;
        dcbSerialParams.ByteSize = 8;
        dcbSerialParams.Parity = NOPARITY;
        dcbSerialParams.StopBits = ONESTOPBIT;

        SetCommState(hSerial, &dcbSerialParams);
    }
}

```

```

}
else
{
    if (GetLastError() == ERROR_FILE_NOT_FOUND)
    {
        printf("Serial port doesn't exist! \n");
    }

    printf("Error while setting up serial port! \n");
}

```

```
char outputChars[] = "c";
```

```
DWORD btsIOs;
```

```
Mat bottle_saved, hsv_bottle_saved;
```

```
Mat bottle1_saved, hsv_bottle1_saved;
```

```
Mat desabottle_saved, hsv_desabottle_saved;
```

```
Mat desabottle1_saved, hsv_desabottle1_saved;
```

```
Mat desabottle2_saved, hsv_desabottle2_saved;
```

```
Mat can_saved, hsv_can_saved;
```

```
Mat can1_saved, hsv_can1_saved;
```

```
Mat can2_saved, hsv_can2_saved;
```

```
Mat can3_saved, hsv_can3_saved;
```

```
Mat can4_saved, hsv_can4_saved;
```

```
Mat can5_saved, hsv_can5_saved;
```

```
Mat glass_saved, hsv_glass_saved;
```

```
Mat glass1_saved,          hsv_glass1_saved;
Mat glass2_saved,          hsv_glass2_saved;
Mat glass3_saved,          hsv_glass3_saved;

bottle_saved
=imread("C:/Users/Toshiba/Pictures/opencv/bottle/bottle_1.jpg",
CV_LOAD_IMAGE_COLOR);
bottle1_saved
=imread("C:/Users/Toshiba/Pictures/opencv/bottle/bottle_2.jpg",
CV_LOAD_IMAGE_COLOR);
desabottle_saved
=imread("C:/Users/Toshiba/Pictures/opencv/desabottle/desabottle_1.jpg",
CV_LOAD_IMAGE_COLOR);
desabottle1_saved
=imread("C:/Users/Toshiba/Pictures/opencv/desabottle/desabottle_2.jpg",
CV_LOAD_IMAGE_COLOR);
desabottle2_saved
=imread("C:/Users/Toshiba/Pictures/opencv/desabottle/desabottle_3.jpg",
CV_LOAD_IMAGE_COLOR);

can_saved
=imread("C:/Users/Toshiba/Pictures/opencv/milo/milo_1.jpg",
CV_LOAD_IMAGE_COLOR);
can1_saved
=imread("C:/Users/Toshiba/Pictures/opencv/milo/milo_2.jpg",
CV_LOAD_IMAGE_COLOR);
can2_saved
=imread("C:/Users/Toshiba/Pictures/opencv/milo/milo_3.jpg",
CV_LOAD_IMAGE_COLOR);
```

```

can3_saved
=imread("C:/Users/Toshiba/Pictures/opencv/pepsi/pepsi_1.jpg",
CV_LOAD_IMAGE_COLOR);
can4_saved
=imread("C:/Users/Toshiba/Pictures/opencv/pepsi/pepsi_2.jpg",
CV_LOAD_IMAGE_COLOR);
can5_saved
=imread("C:/Users/Toshiba/Pictures/opencv/pepsi/pepsi_3.jpg",
CV_LOAD_IMAGE_COLOR);

```

```

glass_saved
=imread("C:/Users/Toshiba/Pictures/opencv/glass/glass_1.jpg",
CV_LOAD_IMAGE_COLOR);

```

```

glass1_saved
=imread("C:/Users/Toshiba/Pictures/opencv/glass/glass_2.jpg",
CV_LOAD_IMAGE_COLOR);

```

```

glass2_saved
=imread("C:/Users/Toshiba/Pictures/opencv/glass/glass_3.jpg",
CV_LOAD_IMAGE_COLOR);

```

```

glass3_saved
=imread("C:/Users/Toshiba/Pictures/opencv/glass/glass_4.jpg",
CV_LOAD_IMAGE_COLOR);

```

```

cvtColor(bottle_saved,          hsv_bottle_saved,
COLOR_BGR2HSV);
cvtColor(bottle1_saved,       hsv_bottle1_saved,
COLOR_BGR2HSV);
cvtColor(desabottle_saved,    hsv_desabottle_saved,
COLOR_BGR2HSV);

```

```
cvtColor(desabottle1_saved,      hsv_desabottle1_saved,
COLOR_BGR2HSV);
```

```
cvtColor(desabottle2_saved,      hsv_desabottle2_saved,
COLOR_BGR2HSV);
```

```
cvtColor(can_saved,              hsv_can_saved,
COLOR_BGR2HSV);
```

```
cvtColor(can1_saved,             hsv_can1_saved,
COLOR_BGR2HSV);
```

```
cvtColor(can2_saved,             hsv_can2_saved,
COLOR_BGR2HSV);
```

```
cvtColor(can3_saved,             hsv_can3_saved,
COLOR_BGR2HSV);
```

```
cvtColor(can4_saved,             hsv_can4_saved,
COLOR_BGR2HSV);
```

```
cvtColor(can5_saved,             hsv_can5_saved,
COLOR_BGR2HSV);
```

```
cvtColor(glass_saved,            hsv_glass_saved,
COLOR_BGR2HSV);
```

```
cvtColor(glass1_saved,           hsv_glass1_saved,
COLOR_BGR2HSV);
```

```
cvtColor(glass2_saved,           hsv_glass2_saved,
COLOR_BGR2HSV);
```

```
cvtColor(glass3_saved,           hsv_glass3_saved,
COLOR_BGR2HSV);
```

```
int h_value =50;
```

```
int s_value =60;
```

```

int histsize[]={h_value, s_value};

float h_range[]={0,180};
float s_range[]={0,255};
const float* range[]={h_range, s_range};

int channel[]={0,1};

MatND hist_bottle_saved;
MatND hist_bottle1_saved;
MatND hist_desabottle_saved;
MatND hist_desabottle1_saved;
MatND hist_desabottle2_saved;

MatND hist_can_saved;
MatND hist_can1_saved;
MatND hist_can2_saved;
MatND hist_can3_saved;

MatND hist_can4_saved;
MatND hist_can5_saved;

MatND hist_glass_saved;
MatND hist_glass1_saved;
MatND hist_glass2_saved;
MatND hist_glass3_saved;

calcHist(&hsv_bottle_saved, 1, channel, Mat(), hist_bottle_saved, 2, histsize,
range, true, false);

```

```

normalize(hist_bottle_saved, hist_bottle_saved, 0, 1, NORM_MINMAX, -1,
Mat());
calcHist(&hsv_bottle1_saved, 1, channel, Mat(), hist_bottle1_saved, 2, histsize,
range, true, false);
normalize(hist_bottle1_saved, hist_bottle1_saved, 0, 1, NORM_MINMAX, -1,
Mat());

calcHist(&hsv_desabottle_saved, 1, channel, Mat(), hist_desabottle_saved, 2,
histsize, range, true, false);
normalize(hist_desabottle_saved, hist_desabottle_saved, 0, 1, NORM_MINMAX,
-1, Mat());
calcHist(&hsv_desabottle1_saved, 1, channel, Mat(), hist_desabottle1_saved, 2,
histsize, range, true, false);
normalize(hist_desabottle1_saved, hist_desabottle1_saved, 0, 1,
NORM_MINMAX, -1, Mat());
calcHist(&hsv_desabottle2_saved, 2, channel, Mat(), hist_desabottle2_saved, 2,
histsize, range, true, false);
normalize(hist_desabottle2_saved, hist_desabottle2_saved, 0, 1,
NORM_MINMAX, -1, Mat());

calcHist(&hsv_can_saved, 1, channel, Mat(), hist_can_saved, 2, histsize, range,
true, false);
normalize(hist_can_saved, hist_can_saved, 0, 1, NORM_MINMAX, -1, Mat());
calcHist(&hsv_can1_saved, 1, channel, Mat(), hist_can1_saved, 2, histsize, range,
true, false);
normalize(hist_can1_saved, hist_can1_saved, 0, 1, NORM_MINMAX, -1,
Mat());
calcHist(&hsv_can2_saved, 1, channel, Mat(), hist_can2_saved, 2, histsize, range,
true, false);

```

```

normalize(hist_can2_saved, hist_can2_saved, 0, 1, NORM_MINMAX, -1,
Mat());
calcHist(&hsv_can3_saved, 1, channel, Mat(), hist_can3_saved, 2, histsize, range,
true, false);
normalize(hist_can3_saved, hist_can3_saved, 0, 1, NORM_MINMAX, -1,
Mat());

calcHist(&hsv_can4_saved, 1, channel, Mat(), hist_can4_saved, 2, histsize, range,
true, false);
normalize(hist_can4_saved, hist_can4_saved, 0, 1, NORM_MINMAX, -1,
Mat());
calcHist(&hsv_can5_saved, 1, channel, Mat(), hist_can5_saved, 2, histsize, range,
true, false);
normalize(hist_can5_saved, hist_can5_saved, 0, 1, NORM_MINMAX, -1,
Mat());

calcHist(&hsv_glass_saved, 1, channel, Mat(), hist_glass_saved, 2, histsize,
range, true, false);
normalize(hist_glass_saved, hist_glass_saved, 0, 1, NORM_MINMAX, -1,
Mat());
calcHist(&hsv_glass1_saved, 1, channel, Mat(), hist_glass1_saved, 2, histsize,
range, true, false);
normalize(hist_glass1_saved, hist_glass1_saved, 0, 1, NORM_MINMAX, -1,
Mat());
calcHist(&hsv_glass2_saved, 1, channel, Mat(), hist_glass2_saved, 2, histsize,
range, true, false);
normalize(hist_glass2_saved, hist_glass2_saved, 0, 1, NORM_MINMAX, -1,
Mat());
calcHist(&hsv_glass3_saved, 1, channel, Mat(), hist_glass3_saved, 2, histsize,
range, true, false);

```



```

    normalize(hist_class3_saved, hist_class3_saved, 0, 1, NORM_MINMAX, -1,
Mat());

```

```

VideoCapture cap(1);
    while (1)
    {
        Mat frame;

        bool bSuccess = cap.read(frame); // read a new frame from video

        if (!bSuccess) //if not success, break loop
        {
            cout << "Cannot read a frame from video stream" << endl;
            break;
        }
        if (waitKey(30) == 27) //wait for 'esc' key press for 30ms. If 'esc' key is pressed,
break loop
        {
            cout << "esc key is pressed by user" << endl;
            break;
        }

        cap >> frame;
        //imshow("Recycle Object View", frame);
        imwrite("C:/Users/Toshiba/Pictures/opencvtest1.jpg", frame);

        Mat cam_input,          hsv_cam_input;

        cam_input          =imread("C:/Users/Toshiba/Pictures/opencvtest1.jpg",
CV_LOAD_IMAGE_COLOR);

```

```

cvtColor(cam_input,          hsv_cam_input,
COLOR_BGR2HSV);

MatND hist_cam_input;

calcHist(&hsv_cam_input, 1, channel, Mat(), hist_cam_input, 2, histsize, range,
true, false);

normalize(hist_cam_input, hist_cam_input, 0, 1, NORM_MINMAX, -1, Mat());

double bottle_input = compareHist(hist_bottle_saved, hist_cam_input, 0);
double bottle1_input = compareHist(hist_bottle1_saved, hist_cam_input, 0);
double desabottle_input = compareHist(hist_desabottle_saved, hist_cam_input,
0);
double desabottle1_input = compareHist(hist_desabottle1_saved, hist_cam_input,
0);
double desabottle2_input = compareHist(hist_desabottle2_saved, hist_cam_input,
0);
double glass_input = compareHist(hist_glass_saved, hist_cam_input, 0);
double glass1_input = compareHist(hist_glass1_saved, hist_cam_input, 0);
double glass2_input = compareHist(hist_glass2_saved, hist_cam_input, 0);
double glass3_input = compareHist(hist_glass3_saved, hist_cam_input, 0);

double can_input = compareHist(hist_can_saved, hist_cam_input, 0);
double can1_input = compareHist(hist_can1_saved, hist_cam_input, 0);
double can2_input = compareHist(hist_can2_saved, hist_cam_input, 0);
double can3_input = compareHist(hist_can3_saved, hist_cam_input, 0);

double can4_input = compareHist(hist_can4_saved, hist_cam_input, 0);
double can5_input = compareHist(hist_can5_saved, hist_cam_input, 0);

```

```

if(bottle_input>=0.8 || bottle1_input>=0.8 || desabottle_input>=0.76 ||
desabottle1_input>=0.76 || desabottle2_input>=0.8)
{
    outputChars[0] = 'A';
    WriteFile(hSerial, outputChars, strlen(outputChars), &btsIOs, NULL);
    printf("It is a plastic Bottle\n");
    cout<<"Comparison (1) is " <<bottle_input<<endl;
    cout<<"Comparison (1.1) is " <<bottle1_input<<endl;
    cout<<"Comparison (2) is " <<desabottle_input<<endl;
    cout<<"Comparison (3) is " <<desabottle1_input<<endl;
    cout<<"Comparison (4) is " <<desabottle2_input<<endl;
    ofstream myfile;
    myfile.open ("TEXT.txt",ios::app);
    myfile << "It is a plastic bottle.\n";
    myfile << "Comparison (1) is " <<bottle_input<<endl;
    myfile << "Comparison (1.1) is " <<bottle1_input<<endl;
    myfile << "Comparison (2) is " <<desabottle_input<<endl;
    myfile << "Comparison (3) is " <<desabottle1_input<<endl;
    myfile << "Comparison (4) is " <<desabottle2_input<<endl<<endl;
    myfile.close();
    Sleep(20000);
    //system("pause");
}

```

```

if(glass_input>=0.72 || glass1_input>=0.72 || glass2_input>=0.73 ||
glass3_input>=0.78)
{
    outputChars[0] = 'B';
    WriteFile(hSerial, outputChars, strlen(outputChars), &btsIOs, NULL);
    printf("It is a glass Bottle!!!\n");
    cout<<"Comparison (1) is " <<glass_input<<endl;

```

```

cout<<"Comparison (2) is "<<glass1_input<<endl;
cout<<"Comparison (3) is "<<glass2_input<<endl;
cout<<"Comparison (4) is "<<glass3_input<<endl;
ofstream myfile;
myfile.open ("TEXT.txt",ios::app);
myfile <<"It is a glass bottle."<<endl;
myfile << "Comparison (1) is "<<glass_input<<endl;
myfile << "Comparison (2) is "<<glass1_input<<endl;
myfile << "Comparison (3) is "<<glass2_input<<endl;
myfile << "Comparison (4) is "<<glass3_input<<endl<<endl;
myfile.close();
Sleep(18000);
//system("pause");
}

if( can_input>=0.68 || can1_input>=0.68 || can2_input>=0.68 || can3_input>=0.68
||can4_input>=0.68 || can5_input>=0.68)
{
    outputChars[0] = 'C';
    WriteFile(hSerial, outputChars, strlen(outputChars), &btsIOs, NULL);
    printf("It is a Can\n");
    cout<<"Comparison (1) is "<<can_input<<endl;
    cout<<"Comparison (2) is "<<can1_input<<endl;
    cout<<"Comparison (3) is "<<can2_input<<endl;
    cout<<"Comparison (4) is "<<can3_input<<endl;
    cout<<"Comparison (5) is "<<can4_input<<endl;
    cout<<"Comparison (6) is "<<can5_input<<endl;
    ofstream myfile;
    myfile.open ("TEXT.txt",ios::app);
    myfile << "It is a Can.\n";
    myfile << "Comparison (1) is "<<can_input<<endl;

```

```

myfile << "Comparison (2) is " << can1_input << endl;
myfile << "Comparison (3) is " << can2_input << endl;
myfile << "Comparison (4) is " << can3_input << endl;
myfile << "Comparison (5) is " << can4_input << endl;
myfile << "Comparison (6) is " << can5_input << endl << endl;
myfile.close();
Sleep(20000);
//system("pause");
}

else
{
    printf("Scanning\n");
    cout << " " << glass_input << " " << glass1_input << " " << glass2_input
<< " " << glass3_input << endl;
    cout << " " << bottle_input << " " << bottle1_input << " " << desabottle_input
<< " " << desabottle1_input << " " << desabottle2_input << endl;
    cout << " " << can_input << " " << can1_input << " " << can2_input
<< " " << can3_input << " " << can4_input
<< " " << can5_input << endl;

    Sleep(3500);
    //system("pause");
}

}

return 0;

```

### 7.3 Coding for Arduino output

```

#include <Servo.h>

Servo servogripper;

Servo myservo;

String inputString = "";

String outputString = "";

boolean stringComplete = false;

int led_bottle=13;

int led_glass=3;

int led_can=2;

void setup() {

  Serial.begin(9600);

  servogripper.attach(9); //gripper

  myservo.attach(10); //rotate

  inputString.reserve(256);

  outputString.reserve(256);

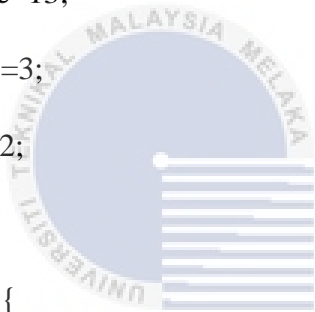
  pinMode(led_bottle, OUTPUT);

  pinMode(led_glass, OUTPUT);

  pinMode(led_can, OUTPUT);

  servogripper.write(0);

```



اونيورسي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

```

delay(3000);
}

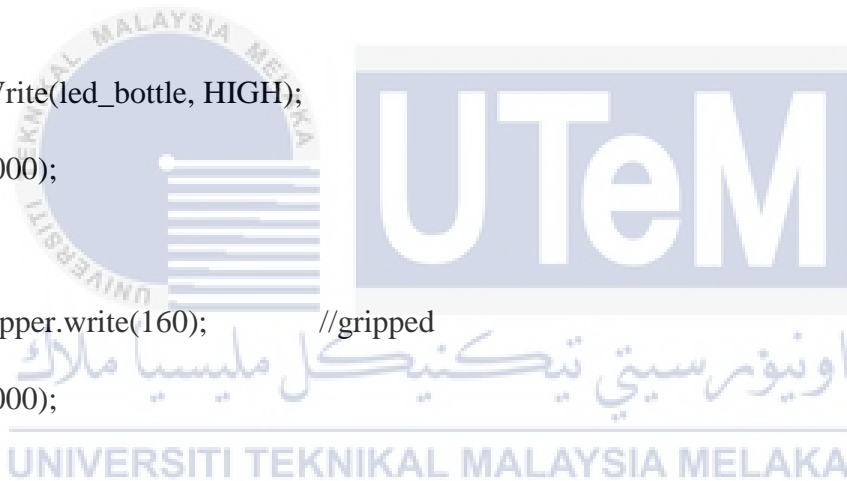
void loop()
{
  if (stringComplete)
  {
    if(inputString.equals("A"))      //plastic bottle
    {
      digitalWrite(led_bottle, HIGH);
      delay(1000);
      servogripper.write(160);      //gripped
      delay(2000);

      myservo.write(45);           //rotate clockwise to plastic station
      delay(600);

      myservo.write(89);          //stop
      delay(2000);

      servogripper.write(0);      //gripper release, drop to can station
      delay(3000);

```



```
myservo.write(110);          //rotate anti-clockwise

delay(590);

myservo.write(89);          //stop

delay(2000);

digitalWrite(led_bottle,LOW);

delay(10);

outputString=inputString;
}

if(inputString.equals("B")) //glass bottle
{
digitalWrite(led_glass, HIGH);
delay(1000);

servogripper.write(160);    //gripped


delay(2000);

myservo.write(45);          //rotate clockwise to can station

delay(800);

myservo.write(89);          //stop

delay(2000);
```





```

servogripper.write(0);      //gripper release, drop to can station
delay(3000);

myservo.write(110);        //rotate anti-clockwise
delay(790);

myservo.write(89);        //stop
delay(2000);

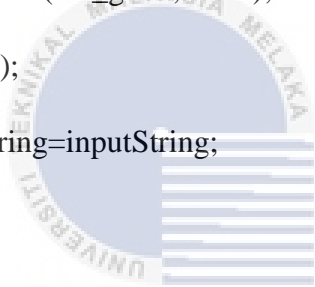
digitalWrite(led_glass,LOW);
delay(10);
outputString=inputString;
}

if(inputString.equals("C")) //aluminium can
{
    digitalWrite(led_can, HIGH);
    delay(1000);

    servogripper.write(160);    //gripped
    delay(2000);

    myservo.write(45);        //rotate clockwise to can station
    delay(1000);

```



اونيورسي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

```

myservo.write(89);          //stop

delay(2000);

servogripper.write(0);     //gripper release, drop to can station

delay(3000);

myservo.write(110);       //rotate anti-clockwise

delay(990);

myservo.write(89);        //stop
delay(2000);

digitalWrite(led_can,LOW);

delay(10);
outputString=inputString;
}

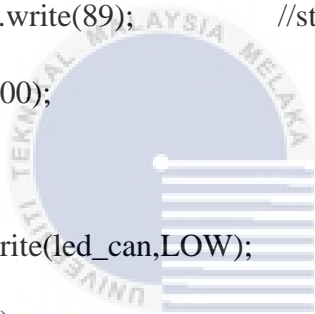
if(!(inputString.equals("A")||inputString.equals("B")||inputString.equals("C")))
{
    inputString.setCharAt(0,'-');

    outputString=inputString;

    delay(10);
}

if(Serial.available())
{

```



اونيورسي تيكنيكل مليسيا ملاك  
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

```
Serial.println(outputString);  
  
//Serial.flush();  
  
}  
  
delay(10);  
  
// clear the string:  
  
inputString = "";  
  
stringComplete = false;  
  
}  
  
}  
  
void serialEvent()  
{  
    inputString="";  
    while (Serial.available())  
    {  
        // get the new byte:  
  
        char inChar = (char)Serial.read();  
  
        inputString += inChar;  
  
        stringComplete = true;  
  
        inputString += '\0';  
  
    }  
}
```



اونيورسي تيكنيكل مليسيا ملاك  
UNIVERSITI TEKNIKAL MALAYSIA MELAKA



### 7.5 Databased image



bottle\_1



bottle\_2



desabottle\_1



desabottle\_2



desabottle\_3



glass\_1



glass\_2



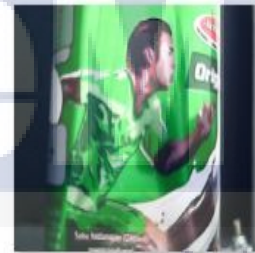
glass\_3



glass\_4



milo\_1



milo\_2



milo\_3



milo\_4



pepsi\_1



pepsi\_2



pepsi\_3