

# QUALITY CONTROL SYSTEM



BY

CQMS MUHAMMAD NASIR NADEEM

GC MUSTAFIZ UR REHMAN

GC WALEED UMER

FC MUFTAH ABU HASENA

Submitted to the Faculty of Department of Electrical Engineering,  
Military College of Signals, National University of Sciences and Technology,  
Islamabad in partial fulfilment for the requirements of a B.E Degree in  
Telecom Engineering

June 2017

# ABSTRACT

---

## QUALITY CONTROL SYSTEM

Quality Control is a foremost part of any production line and pharmaceutical companies are no omission to this. An automated Quality Control System is essential to make any production line economically practicable, as manual labour is significantly cost ineffective. Creating an automated Quality Control System to identify defective medicine packaging is the prime objective of this project. The developed system will be responsible for sensing sub-standard medicine packaging and taking them out of the production line.

We have tried to solve this problem by creating a Quality Control System that senses faulty medicine packaging through Image Processing Techniques. The system will identify any leaflet of tablets that does not happen to meet the set requirements and a mechanical pusher will do away with that package from the production line.

This report contains narrative description about the project along with design details both software and hardware, project analysis and evaluation, results of rigorous testing and their analysis, complications confronted while development of the project, a demonstration outline and recommendations for future work.

# Certificate

It is hereby certified that the contents of this project entitled “Automated Quality Control System” have been found satisfactory as per the requirements of the B.E. Electrical (Telecom) Engineering Degree.

Project Supervisor  
Asst Prof Dr. Mir Yasir Umair  
MCS, NUST

# Copyright Notice

Copyright in script of this thesis is retained by the student authors, copies either in full or of portions, may be made only by the permission of the authors and lodged in the Library of Military College of Signals, NUST. Details may be obtained by the Library. Further copies (by any process) made in accordance with such instructions may not be made without the permission (in writing) of the authors.

# Dedication

*Dedicated to*

*Allah Almighty*

*Dr Mir Yasir*

*Our families for their confidence in us*

# Acknowledgement

First, we would thank ALLAH ALMIGHTY who enlightened us with the knowledge that we needed for our project and gave us strength to accomplish our task successfully.

We acknowledge, with profound gratefulness and indebtedness, the motivation, backing and unceasing guidance provided to us by Dr. Mir Yasir Umair for our project.

We are grateful to our colleagues and to the people who have been readily available and willing to help us in developing the project.

Special thanks to our families for their continuous support and motivation throughout the process.

# Table of Contents

<b>1. Introduction</b>	<b>1</b>
<b>1.1. Background/Motivation</b>	<b>1</b>
<b>1.2. Project Description</b>	<b>2</b>
<b>1.3. Salient Features</b>	<b>2</b>
<b>1.4. Scope</b>	<b>3</b>
<b>1.5. Organization of document</b>	<b>4</b>
<b>1.6. Sequential Diagram</b>	<b>4</b>
<b>2. Background Study</b>	<b>5</b>
<b>2.1. Use of Background Study</b>	<b>7</b>
<b>3. Design and Development</b>	<b>8</b>
<b>3.1. Development</b>	<b>8</b>
<b>3.2. Design Detail</b>	<b>8</b>
3.2.1. Conveyor Belt	8
3.2.2. Detection	9
3.2.3. Conversion from RGB to Binary	9
<b>3.3. Hough Circle Transform</b>	<b>11</b>
3.3.1. Search with Fixed R	11
3.3.2. Multiple Circles with known Radius	12
<b>3.4. Blob Detection</b>	<b>13</b>
3.4.1. Detection Procedure	13
3.4.2. Detection by shape, colour and size	14
<b>3.5. OpenCV</b>	<b>16</b>
<b>4. Hardware Design</b>	<b>17</b>
<b>4.1. Camera</b>	<b>17</b>
4.1.1. Specifications	18
<b>4.2. Raspberry pi 2 Board:</b>	<b>18</b>
4.2.1. Specifications:	19
<b>4.3. ATMEGA 328 PU IC</b>	<b>20</b>
<b>4.4. Bluetooth Module</b>	<b>21</b>
<b>4.5. Mechanical Pusher</b>	<b>22</b>

4.5.1.	Elements:.....	22
4.5.2.	Construction:.....	22
4.5.3.	Disk:.....	22
4.5.4.	Stepper motor:.....	23
4.5.5.	Plastic Rod and Pusher:.....	24
<b>5.</b>	<b>Software Design</b> .....	<b>25</b>
5.1.	Android Application .....	25
5.2.	Technique for Image Processing: .....	28
5.3.	Explanation: .....	28
5.3.1.	Detection of faulty leaflets:.....	28
5.3.2.	Stepper Motor: .....	28
5.4.	Project Analysis and Evaluation.....	29
5.4.1.	Simulation .....	30
<b>6.</b>	<b>Future Work</b> .....	<b>38</b>
<b>7.</b>	<b>Conclusion</b> .....	<b>39</b>
7.1	Summary.....	39
7.2	Objectives Accomplished.....	39
7.3	Accomplishments .....	39
<b>8.</b>	<b>Bibliography</b> .....	<b>40</b>
<b>9.</b>	<b>Appendix A</b> .....	<b>42</b>
<b>10.</b>	<b>Appendix B</b> .....	<b>43</b>
	ANDROID APPLICATION CODE.....	43
	IMAGE PROCESSING CODE .....	54
	STEPPER MOTOR CODE .....	59
<b>11.</b>	<b>Appendix C</b> .....	<b>62</b>
	DETAILED FLOW CHART .....	62
<b>12.</b>	<b>Appendix D</b> .....	<b>63</b>



# Table of Figures

Figure 1 Types of medicine.....	03
Figure 2 Sequential diagram.....	04
Figure 3 Project Development.....	08
Figure 4 Objects to be detected.....	09
Figure 5 Conversion to binary.....	10
Figure 6 Boundary detection.....	10
Figure 7 Highlighting single boundary.....	12
Figure 8 Highlighting multiple boundary.....	12
Figure 9 Dark regions representing Blobs.....	13
Figure 10 Different Blob detection types.....	14
Figure 11 Blob detection by Convexity.....	15
Figure 12 Blob detection by Inertia Ratio.....	15
Figure 13 Components of Raspberry Pi camera.....	17
Figure 14 Raspberry Pi 2 Model B+ Port Label.....	19
Figure 15 ATMEL ATMEGA 328-PU.....	20
Figure 16 Pin configuration of ATMEGA 328 PU.....	21
Figure 17 Bluetooth Module HC-05.....	22
Figure 18 Stepper Motor.....	23
Figure 19 Motor Driver (L298n).....	24
Figure 20 Interface screen 1 of Pi Medicine.....	26
Figure 21 Interface screen 2 of Pi Medicine.....	26
Figure 22 SMS being sent.....	27
Figure 23 SMS received.....	27
Figure 24 Simulation result (example 1).....	29
Figure 25 Function call in Command Prompt (Raspberry Pi).....	30
Figure 26 Python code.....	31
Figure 27 Simulation result (medicine 1).....	32
Figure 28 Output image of faultless medicine 1 leaflet.....	32
Figure 29 Output image of defective medicine 1 leaflet.....	33
Figure 30 Medicine 2.....	34
Figure 31 Simulation result (medicine 2).....	34
Figure 32 Medicine 3.....	35

Figure 33 Simulation result (medicine 3).....	35
Figure 34 Output image of faultless medicine 3 leaflet.....	36
Figure 35 Output image of defective medicine 3 leaflet.....	36
Figure 36 Leaflet containing broken capsules.....	37

# List of Abbreviations

<b>SMS:</b>	Short Message Service
<b>MATLAB:</b>	Matrix Laboratory
<b>RGB:</b>	Red Green Blue
<b>HSI:</b>	Hue Saturation and Intensity
<b>R:</b>	Radius
<b>RAM:</b>	Random Access Memory
<b>ROM:</b>	Read Only Memory
<b>USB:</b>	Universal Serial Bus
<b>HD:</b>	High Definition
<b>HDMI:</b>	High-Definition Multimedia Interface
<b>I/O:</b>	Input/output
<b>GPIO:</b>	General Purpose Input/Output
<b>PWM:</b>	Pulse Width Modulation
<b>MP:</b>	Mega Pixels
<b>LED:</b>	Light Emitting Diode
<b>DC:</b>	Direct Current
<b>MIPS:</b>	Million Instruction per Second
<b>App:</b>	Application
<b>MIPI:</b>	Mobile Industry Processor Interface
<b>CSI:</b>	Camera Serial Interface
<b>CMOS:</b>	Complementary Metal Oxide Semiconductor
<b>SPP:</b>	Serial Port Protocol
<b>EDR:</b>	Enhanced Data Rate

# 1. Introduction

---

The Medicine manufacturing Industries have a very crucial job in the society. This industry is responsible for providing the community with means to cater health problems. The nature of their work demands an immense need for the quality assurance of the products. Presently most industries are using manual labour to ensure that faulty medicine packets are not passed onto the market. There are generally 4 to 6 persons per production line, controlling and maintaining the standardized quality. The need to automate this practice was felt.

## 1.1. Background/Motivation

Many glitches were found to be supplemented with labour-intensive packaging. Firstly, it is very costly particularly when compared with an automated system. The firms could lessen the burden onto their finances by dropping the salaries of labours. They need to pay only one time procurement fee for an automated system rather than catering for salaries on periodic basis.

Secondly, the effectiveness of such a process is uncertain. Humans are very much susceptible to fault, a point which has been emphasized by science and corrected by the advancement in technology. The efficiency is certain to increase using an automated system.

Then there are apprehensions concerning hygiene. Having people handle the leaflets before they are completely packed is a chancy method. If the hygiene of the workers is not kept under strict examination, there is a risk of the medicine getting contaminated which would setback the prime purpose of the medicine.

## 1.2. Project Description

This project is primarily based on image processing and signal processing; so by this project we tend to incorporate our theoretical knowledge with practical application to gain further understanding in the field of electrical engineering and enhance our skills in the fields of wireless communication and signal processing.

This project is responsible to correctly detect defective packets in a production line. After detecting the defective packet, it has to remove that very packet from the production line in order to stop it from being completely packed so that it is not approved for entry into the market. When a defective leaflet is detected, a mechanical pusher mounted at the end of the production line is responsible for doing away with that leaflet from the production line.

## 1.3. Salient Features

The system consists of five main parts:

- Choosing type of medicine from android app
- Capturing Image of the leaflet
- Use of Image Processing Techniques to detect if the leaflet is defective
- Removal of defective packet using a mechanical pusher
- Intimation, if faulty leaflets are in bulk

A camera will be used to capture images of each passing medicine leaflet. Type of medicine will be selected from android application. The packets will be set on a moving conveyer belt. When a medicine packet will come under the camera, an image will be captured and sent for processing to the Raspberry Pi processor. Using certain image processing techniques, defects in the medicine packets will be identified. A mechanical pusher will be allocated the responsibility of removing the defective leaflets from the conveyor belt. Approved leaflets will be forwarded for final packaging. Too much faulty medicine in bulk will be alerted through an sms or notification over android application.

This project is capable of detecting more than one type of medicine. The medicines which can be detected using this project are given below.

Sr.no	Type of leaflet	Characteristics	Things to be Detected
1	Medicine 1 (Panadol)	1. Shape dependent 2. Colour dependent	1. Round circle detection 2. Colour other than white
2	Medicine 2 (Artifin)	1. Shape dependent 2. Colour Independent	1. Round circle detection
3	Medicine 3 (Nospa)	1. Shape Independent 2. Colour Independent	1. Cylindrical shape detection

Figure 1: Types of medicine

## 1.4. Scope

This project is quality control system for pharmaceutical industries. Tablet packets are examined before packing by an automated process which is less susceptible to errors and inaccuracies. The project should identify any defective leaflet and then remove it from the production line.

The processor being employed for the project is a raspberry pi board. The coding is done in Python using the OpenCV (Open source computer vision Library). Hough Circle transform will be applied to sense the circles. When a defective leaflet is identified, a signal will be sent by the raspberry pi to the mechanical pusher to push the leaflet out of the production line. Notification by android app or sms will be send by android app if the number of defective leaflets cross a pre-set threshold.

This project is helping us to polish and refine the knowledge that we have been able to gain through the course of the degree to practical use and increase our expertise in the fields of Image Processing, Signal Processing, Wireless Communications, Embedded Systems, Networking and Programming.

## 1.5. Organization of document

This document is divided into four main parts:

- Introduction
- Previous work in the field, and how we have taken use of it into our project as per our needs.
- Design, Specifications and Algorithm.
- References and Bibliography.

## 1.6. Sequential Diagram

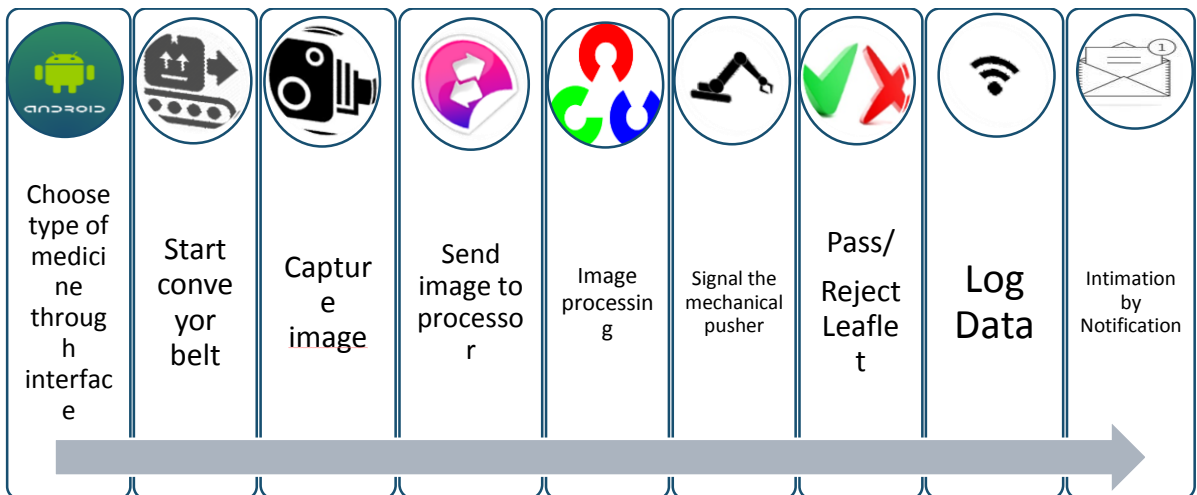


Figure 2: Sequential Diagram

## 2. Background Study

---

Koneru Lakshmaiah Education Foundation published a research paper that focuses on "the design and selection of the proper conveyor belt". This paper provides information regarding the design of conveyor system used, which include speed of belt, width of belt, selection of motor, specifications of belt, diameter of shaft, pulley, gear box selection, with the help of standard model calculation. [1]

A research paper published by M. Khojastehnazhand, M. Omid and A. Tabatabaeefar about "Development of a lemon sorting system based on color and size" talks about the classification of citrus fruits which include oranges, lemons and tangerines. There are two stages of inspection in the sorting system; external inspection and internal inspection. The external inspection is done by the process of thermal imaging while the internal inspection needs certain sensors for the contents of moisture and acid. In this research paper, an effective method for sorting citrus fruits is designed and materialized in visual basic environment. The system consists of two cameras, two capture cards, an appropriate lighting system, a processing unit and other mechanical components. After capturing the image, this technique first gets the fruit out from the background. The specimen of various categories of citrus fruits are captured by the cameras and the data on the HSI color values and estimated volumes of fruits are distilled and stored in a database beforehand. By matching the data on the captured image during sorting phase with the already stored information at the database, the category of the passing fruit is identified. This algorithm can be easily adapted for categorizing, segregating and inspecting other products.

RGB color space is the most approved colour model. Some image processing techniques of fruit vision are based on this colour model (Steinmetz et al., 1996; Leemans et al., 1998; Paulus et al., 1997). When a colour is represented with Red, Green and Blue, the measure of information is multiplied by 3. Hence more designs could be made to thoroughly employ the data in the image. But, this technique is very much prone to lighting and other external effects. Another colour model is HSI (Hue, Saturation and Intensity). Here it is preferred to use HSI, as in this example, Hue value is relatively stable. By computing average Hue (H) value for the fruit, the colour of the passing fruit can be decided. [2]



In another paper object sorting using robotic Pusher based on color detection is designed and implemented. Existing sorting method use a set of different capacitive, inductive, and optical sensors to differentiate object color. In the proposed system a mechatronics color sorting system is developed with the help of image processing technique. Image processing technique senses the objects captured in real-time by a camera and then detects colour and extracts data out of it. In the proposed system a mechatronics color sorting system is developed with the image processing technique. So, the proposed system will eliminate the monotonous work done by human and provides greater accuracy and speed in the work. [3]

From Khulna University, G. M. Atiqur Rahaman and Md. Mobarak Hossain published a research paper "Automatic Defect Detection and Classification Technique from Image: A special case using ceramic tiles". This paper proposes an automatic fault identification and sorting technique that can be employed to ensure better quality of ceramic tiles in production process as well as production rate. This image processing algorithm being used described in the following steps:

- 1<sup>st</sup> step: Executing an image processing technique.
- 2<sup>nd</sup> step: Employ the suggested fault identification process.
- 3<sup>rd</sup> step: Segregating the fault by means of all suggested algorithms.

The procedure adopted, works out a vital role in the ceramic tiles industry, as its role is to identify the faults and manage the quality of ceramic tiles. This automatic segregation process is helpful in regard to procuring information about the pattern of fault in the ceramic tile within a very short time and it also helps to make the decision about the process of recovery so that the faulty pieces do not mix up with the faultless pieces. [4]

Another research paper was published by H. Elbehiery, A. Hefnawy, and M. Elewa on "Surface Defects Detection for Ceramic Tiles Using Image Processing and Morphological Techniques". Their work was a way forward in the field of quality control. They improved the quality by combing the image processing techniques and morphological operation techniques before the final packing of the products to enhance the analogy of packages received by the end users. Images of ceramic tiles were captured by the help of a camera

on the production line at the industry. The image captured was converted to other formats like binary or grey scale to be appropriate for the different fault identification techniques that are used for the different sorts of faults. The paper discusses formal mathematical properties like Edge detection, Morphology operations, Noise reduction, smoothing process, Histogram equalization, and intensity adjustment. By taking a test case tile made of white Plexiglas, various outcomes of lighting and sensitivities were enhanced. [5]

Yet another paper, by Aiyush Suhasaria and Khanindra Pathak, deals with separation of gangue materials from an ore. To extract the required ore from the waste and unrequired substance, Image Processing techniques are employed by live video capturing of the ore and unrequired mixture on the belt. The data captured by camera is sent by any data transferring procedure to a processor. The image processing algorithms are then used to separate the ore and mixture. The output is then used to stimulate certain procedures to direct the mixture towards the outgoing line. Hence, the system lessens the ore reduction and helps in improving the ore quality. The video captured is converted to Hue Saturation Value (HSV) form and apt threshold is applied to separate the ore from the unrequired materials. Sensing by the help of certain threshold in HSV form would take care of the ambient light conditions. The paper provides only a prototype of this technique and gives the output of the software developed for this case in C language. [6]

## **2.1. Use of Background Study**

These research papers are about various image processing techniques and designing of a conveyer belt. Each research paper emphasises on a special aspect of this project. Pros and cons are argued and the most appropriate approach or algorithm is advised. These background studies not only explain the difficulties that occurred for the previous designers but they also propose the enhancements that can be done in the future.

# 3. Design and Development

---

Development and designing are the most important aspects of Final Year Project. This is the part where we take raw knowledge and information from various sources and mould it to suit our requirements.

## 3.1. Development

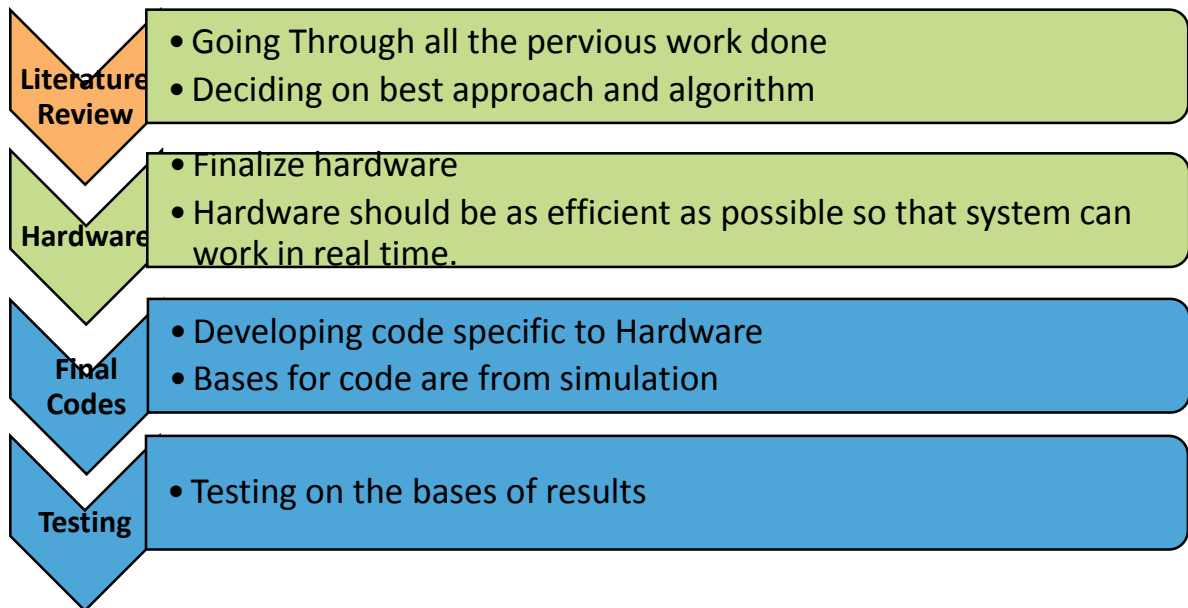


Figure 3: Project Development

## 3.2. Design Detail

### 3.2.1. Conveyor Belt

Conveyor belt is used for carrying objects from one position to another. In this project, Conveyor belt is being used to carry leaflet of tablets over some distance to be examined. The designing of Conveyor belt include speed of belt, selection of motor and belt specifications with the help of standard model calculation.

### 3.2.2. Detection

Image segmentation and object detection is the most vital and significant undertaking of computer vision. This is a critically demanding effort in many applications such as image search and scene understanding etc. Nonetheless it still poses a pertinent issue because of the variety and complexity of object classes and backgrounds.

We will be using Hough transformation algorithm for the detection of circular tablets, which will enable us to identify the defective leaflets.

### 3.2.3. Conversion from RGB to Binary

First we will convert RGB image to a Binary image, so that Hough Transform Circle technique can be applied. After the conversion, we will use Opening and Closing Morphological operation for Noise removal purposes. Boundary detection operation will then follow.

The procedure is diagrammatically illustrated below:



Figure 4: Objects to be detected

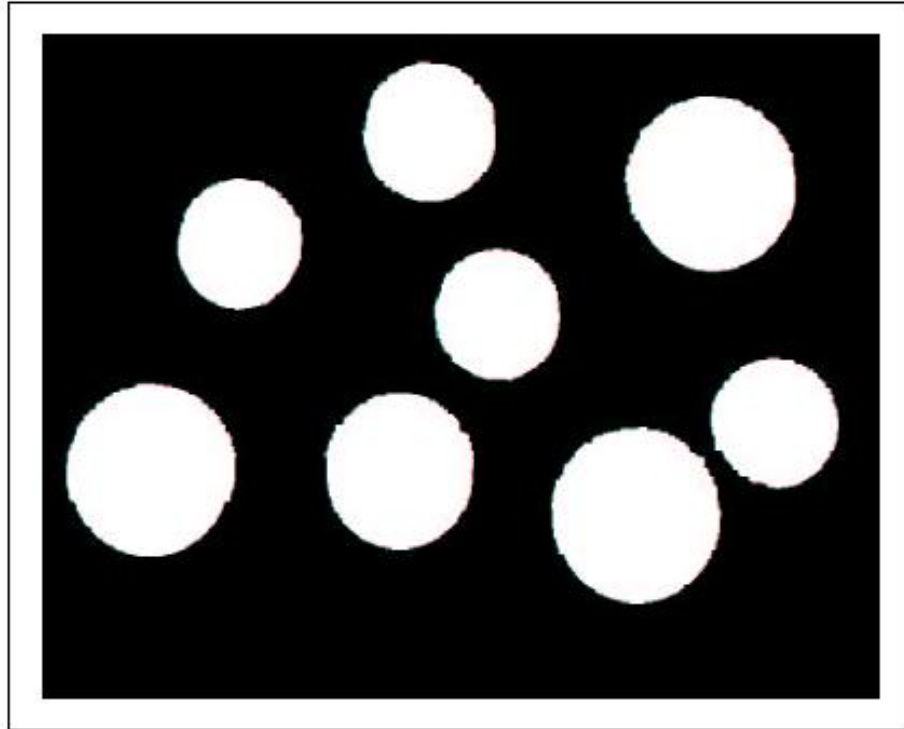


Figure 5: Conversion to Binary

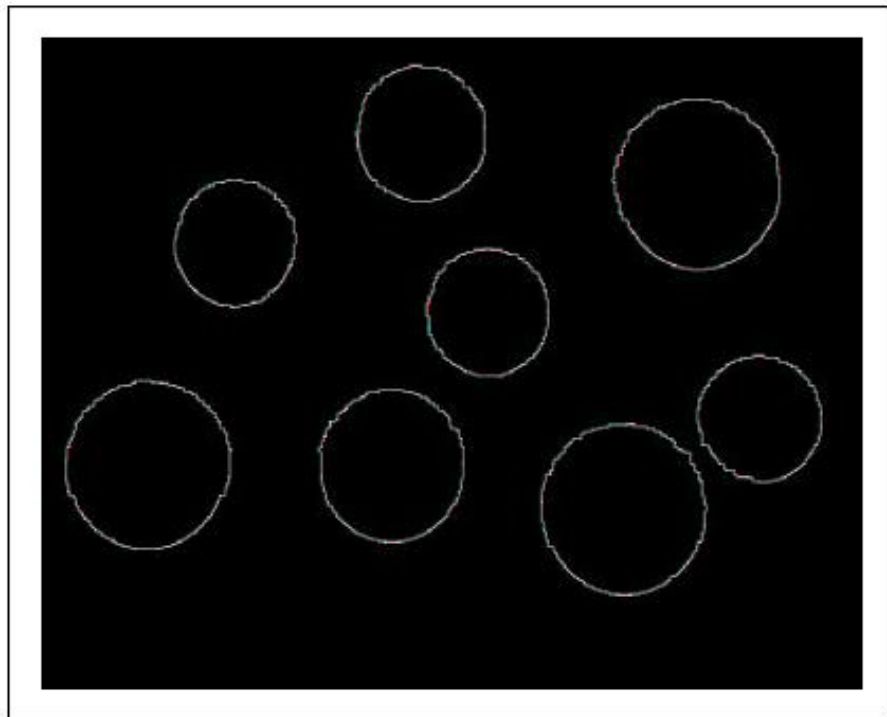


Figure 6: Boundary detection

### 3.3. Hough Circle Transform

Circles are simple geometrical figures of concern in applications of computer vision. This section provides the explanation and demonstration of the use of Hough transform to detect circles.

If the number of points that exist on the circumference of the circle are known, then Hough transform can be employed to define the parameters of the circle. The following parametric equations are used to define a circle having a radius  $R$  and centre at  $(a, b)$ , mathematically.

$$\begin{aligned}x &= a + R \cos(\theta) \\y &= b + R \sin(\theta)\end{aligned}$$

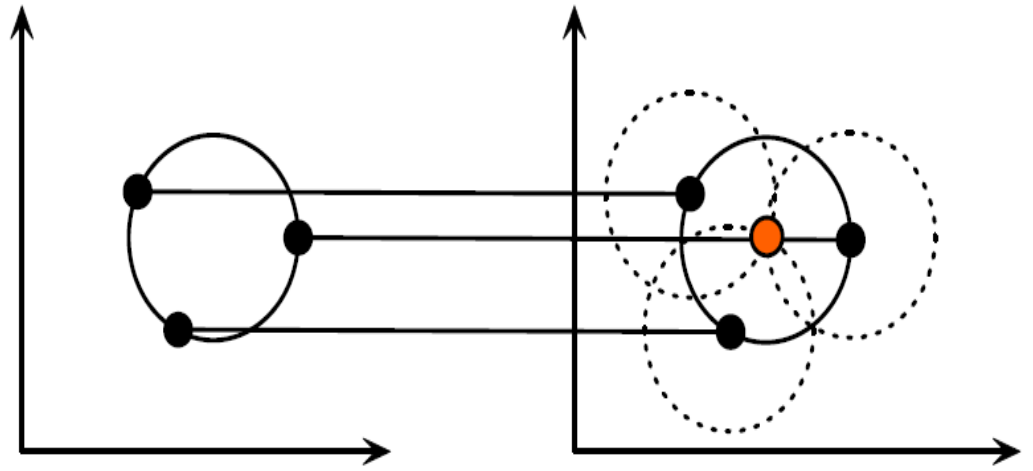
The points  $(x, y)$  indicate the outer circumference of the circle, when the angle  $\theta$  traverses through the complete 360 degree range. If the captured image consists of multiple points and few of them fall on circumference of circles, then each circle is defined by the parametric triplets  $(a, b, R)$  defined by the search program.

#### 3.3.1. Search with Fixed $R$

The objective is to get the coordinates  $(a, b)$  of the circle, so if the radius  $R$  of the circle is known, then the search can be scaled down to 2D.

$$\begin{aligned}x &= a + R \cos(\theta) \\y &= b + R \sin(\theta)\end{aligned}$$

The locus of the point  $(a, b)$  in the parameter space happened to be on a circle having radius  $R$  with centre at  $(x, y)$ . The actual centre point which can be found with a Hough accumulation array will be common to all parameter circles. The following diagram explains the phenomenon taking place.

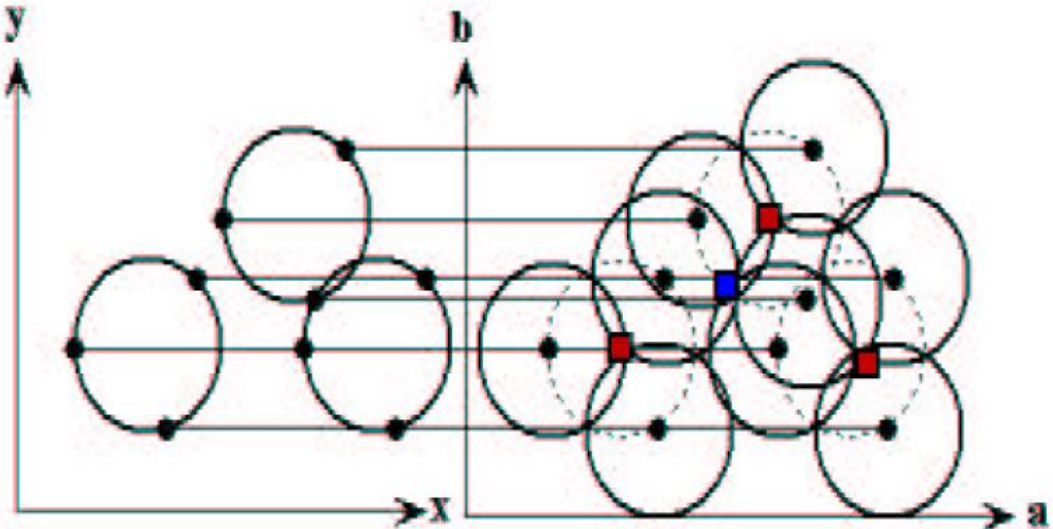


Each point in geometric space (left) generates a circle in parameter space (right). The circles in parameter space intersect at the  $(a, b)$  that is the center in geometric space.

Figure 7: Highlighting single boundary

### 3.3.2. Multiple Circles with known Radius

The same technique described above can be used to find multiple circles with the same radius. The red squares in the parameteric space represent the centre points. Blue squares represent false centres due to the overlapping of the circles.



Each point in geometric space (left) generates a circle in parameter space (right). The circles in parameter space intersect at the  $(a, b)$  that is the center in geometric space.

Figure 8: Highlighting multiple boundary

## 3.4. Blob Detection

A Blob is an accumulation of closely joined pixels in an image that carry some similar features (e.g. grayscale value). In the following image, the dark fields are blobs. The objective of blob detection is to detect and mark these fields as it is done in the following image.

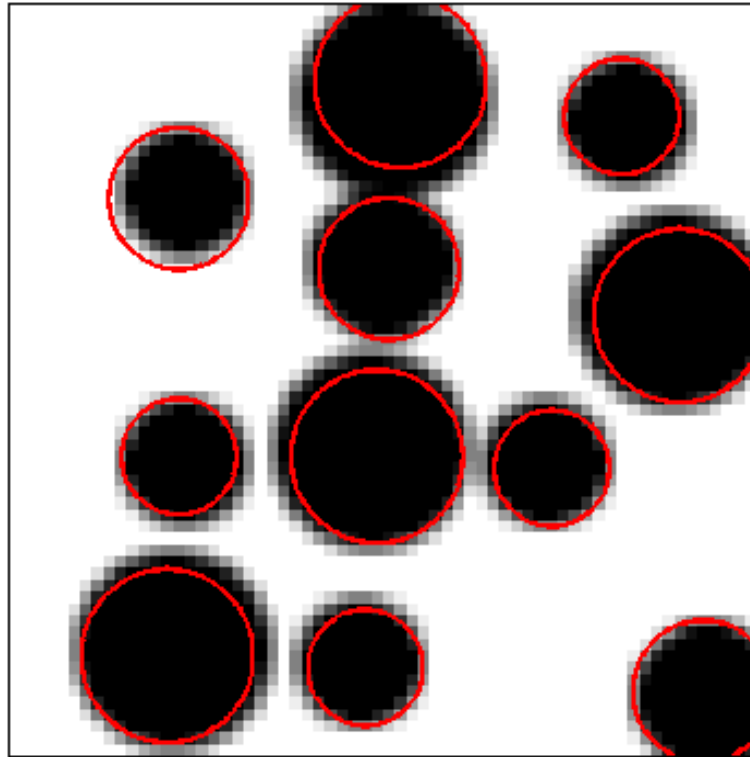


Figure 9: Dark regions representing Blobs

OpenCV brings about an easier method to identify blobs using Python or C++ and filter them based on different characteristics i.e size,color and shape.

### 3.4.1. Detection Procedure

**Thresholding:** Scale the captured image down to multiple binary images. This is achieved by beginning the threshold at `minThreshold`. The final value is `maxThreshold` and `thresholdStep` is used to increase the thresholds. Hence `minThreshold` is the 1<sup>st</sup> value of threshold and in each step `thresholdStep` is added to it. In each increment the `thresholdStep` which is added to `minThreshold` is multiplied by `n`, where `n` is equal to the number of steps. Hence the 2<sup>nd</sup> threshold becomes `minThreshold+2xthresholdStep`. This process is carried on till the time `maxThreshold` is not reached.

**Grouping:** White pixels which are attached with each other are grouped to each other in each binary image. These are referred to as binary blobs.



**Merging:** In each binary image, the center points of the binary blobs are calculated and those binary blobs which are placed close enough are merged together.

**Center and Radius Computation:** After the process of merging, the new blobs emerge. The centers and radii of these merged binary blobs are computed.

### 3.4.2. Detection by shape, colour and size

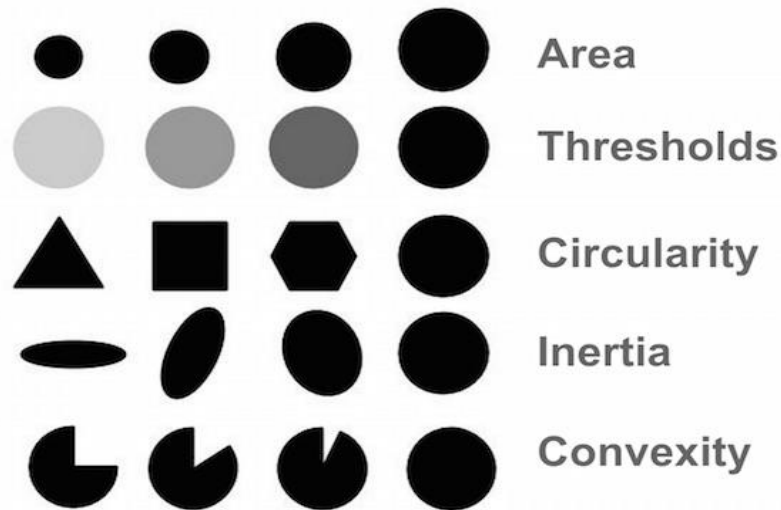


Figure 10: Different Blob detection types

**By Colour:** Filterbycolour is set to 1. For the selection of darker binary blobs, blob colour is set to 0 and for the selection of lighter binary blobs the blob colour is set to 225.

**By Size:** Filterbyarea is set to 1. Then apt values of minimum area and maximum area are set. For example, by setting maxArea=200, all the binary blobs having area more than 200 pixels will be filtered out.

**By Shape:** Shape can be categorized based on three different specifications.

1. **Circularity:** FilterbyCircularity is set to 1. After this apt values of minCircularity and maxCircularity are set. Circularity gives the measure of the shape closer to the circle. For example a square is more circular than a rectangle and a rectangle is more circular than an equilateral triangle. Circularity is defined as

$$\left(\frac{4*\pi*Area}{perimeter * perimeter}\right)$$

A regular circle has a circularity of 1, whereas the circularities of rectangle, square, equilateral triangle and hexagon are 0.698, 0.785, 0.605 and 0.907 respectively.

2. **Convexity:** Convex hull of a shape is defined as the smallest convex part of the shape that fully envelops the shape. Convexity is mathematically given as

$$\text{(Area of the Blob / Area of its convex hull)}$$

FilterbyConvexity is set to 1. After this minimum convexity is set between 0 and 1 and maximum convexity is set to be less than or equal to 1.

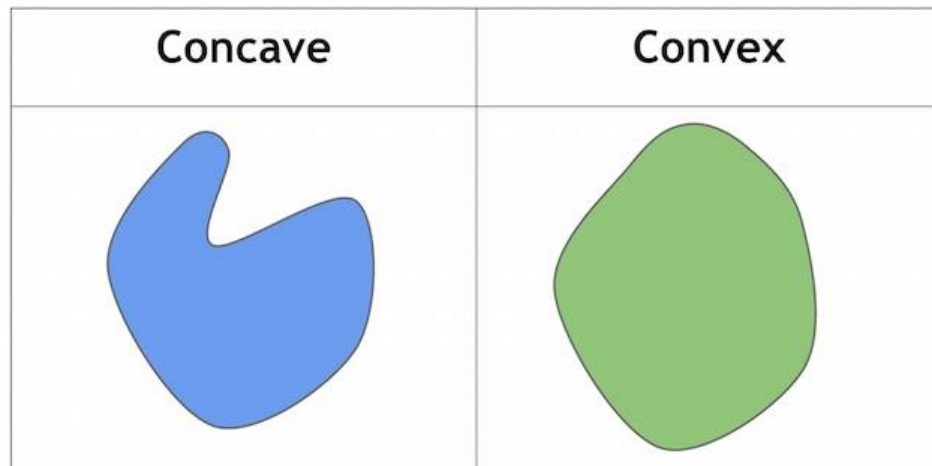


Figure 11: Blob detection By Convexity

3. **Inertia Ratio:** FilterbyInertia is set to 1. Then minimum inertia ratio is set between 0 and 1 and maximum inertia ratio is set to be less than or equal to 1. It is the measure of ellipsity of a shape. For example for a line its value is 0 and for an ellipse its value is between 0 and 1.

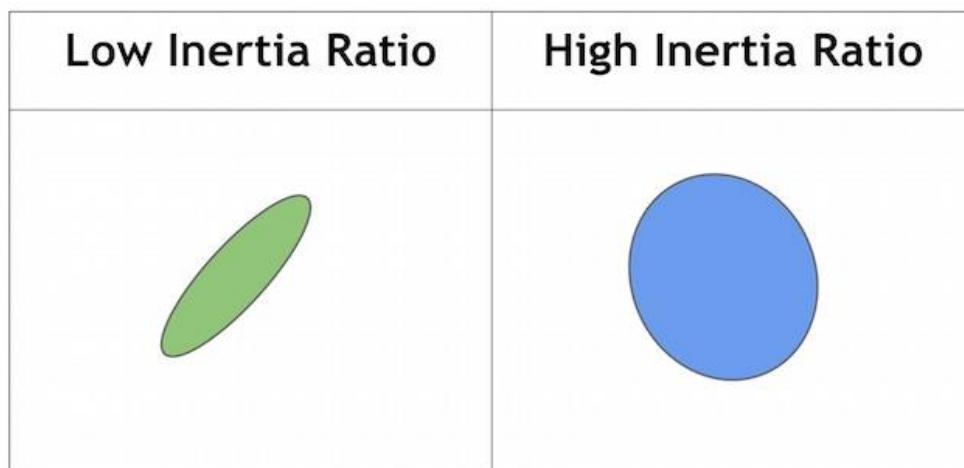


Figure 12: Blob detection By Inertia Ratio

### 3.5. OpenCV

OpenCV (Open Source Computer Vision) is a library of programming functions for real time computer vision. MATLAB is also capable of image processing but in this project Open CV is being used because of the under mentioned reasons.

➤ Speed

MATLAB is based on Java and Java is based on C. Hence, when a MATLAB code is executed, it is turned into Java and then executed. Whereas, OpenCV, as described above is a library of programming functions which is written in C/C++. So it is closer to straight away providing the machine language code to the computer to run the program. It helps getting more image processing done instead of more interpreting. Ultimately the programs run in OpenCV way more fast as compared to those run in MATLAB.

➤ Resources needed

MATLAB is a high level language so it uses relatively more resources of the system. Whereas Open CV requires very less memory. A code written in MATLAB usually requires more than a GB of memory but Open CV code needs only 70 MB.

➤ Portability

Both, MATLAB and OpenCV can run on almost all major operating systems like Windows, Mac OS and Linux. But any device that is capable of running C language can run Open CV. So in this case using Raspberry Pi as the basic image processing system supports the use of OpenCV.

# 4. Hardware Design

---

## 4.1. Camera

Interfacing of camera is one of the most essential constituent of this project. In this part, a Camera is integrated/installed to the Raspberry Pi processor.

The Raspberry Pi processor contains Camera Serial Interface Type 2 (CSI-2) which is specification of Mobile Industry Processor Interface (MIPI). MIPI CSI 2 is an extensively used interface that enables a small camera to be connected to the main Broadcom BCM2835 processor. It is responsible for the provision of an electrical bus connection between the camera and the processor.

The Raspberry Pi Camera Board can be plugged directly into the CSI connector port on the processor. It is capable to capture an image of 5 MP resolution or to record a video of 1080p HD at 30 frames per second. The camera module is attached to the processor by the help of a 15 Pin Ribbon Cable, to the dedicated 15-pin MIPI CSI, which is particularly designed for interfacing of cameras. The MIPI CSI data bus can support extremely high data rates, and it exclusively transfers pixel data to the processor.

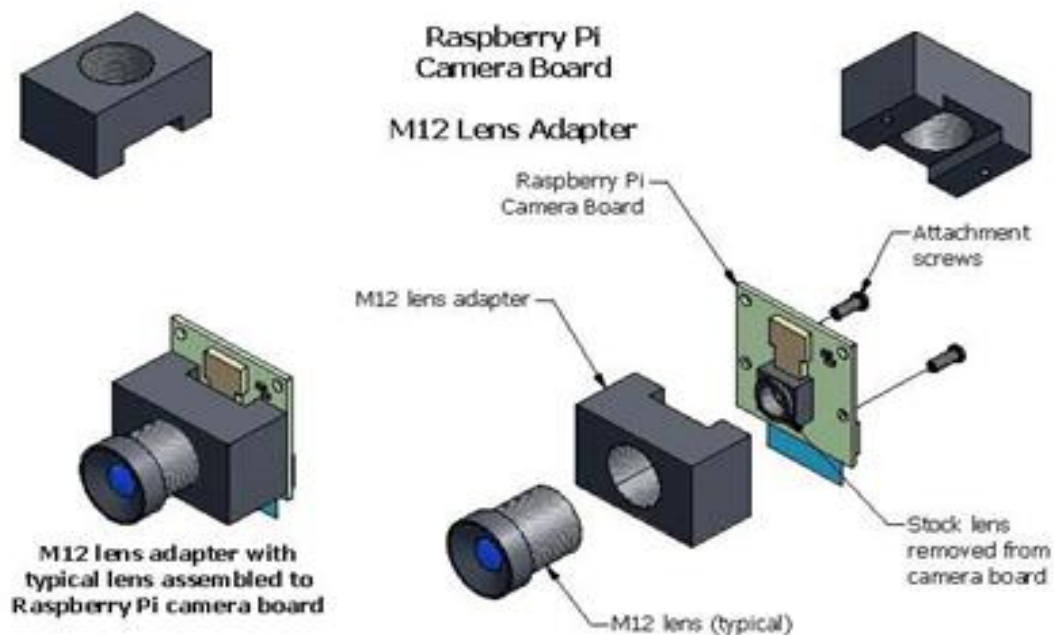


Figure 13: Components of Raspberry Pi camera

### 4.1.1. Specifications

Net price	25 \$
Size	around 25 x 20 x 9 mm
Weight	3 g
Still resolution	5 Megapixels
Video modes	1080p30, 720p60 and 640x480p60/90
Linux integration	V4L2 driver available
C programming API	OpenMAX IL and others available
Sensor	OmniVision OV5647
Sensor resolution	2592 x 1944 pixels
Sensor image area	3.76 x 2.74 mm
Pixel size	1.4 $\mu\text{m}$ x 1.4 $\mu\text{m}$
Optical size	1/4"
Full-frame SLR lens equivalent	35 mm
S/N ratio	36 dB
Dynamic range	67 dB @ 8x gain
Densitivity	680 mV/lux-sec
Dark current	16 mV/sec @ 60 C
Well capacity	4.3 Ke-
Fixed Focus	1 m to infinity
Focal length	3.60 mm +/- 0.01
Horizontal field of view	53.50 +/- 0.13 degrees
Vertical field of view	41.41 +/- 0.11 degress
Focal ratio (F-Stop)	2.9

## 4.2. Raspberry pi 2 Board:

The Raspberry Pi processor is a very low cost, small sized processing unit that can be easily plugged in to a monitor or any HDMI display screen. It is a small device that can be helpful for people of different ages to get into the world of computing. It can help people to develop their programming skills in different programming languages. This small little device is

capable of doing almost every task that can be performed on a desktop or laptop computer, like playing videos, browsing internet, playing video games and as in the case of this project doing real time image processing.

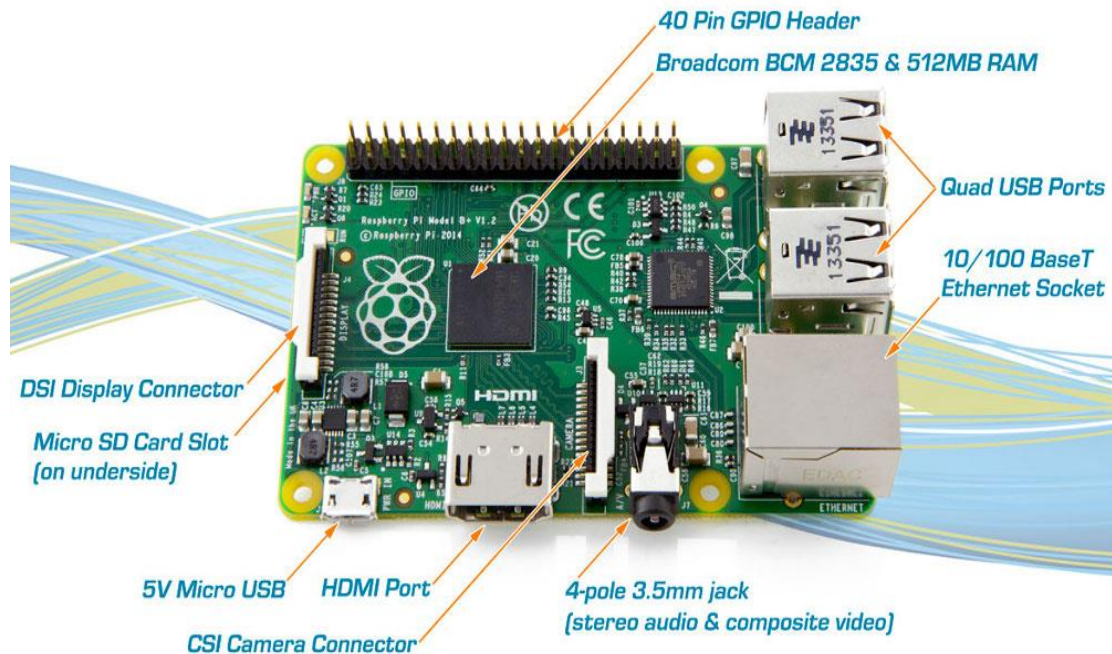


Figure 14: Raspberry Pi 2 Model B+ Port Label

#### 4.2.1. Specifications:

- SoC: Broadcom BCM2836 (CPU, GPU, DSP, SDRAM)
- CPU: 900 MHz quad-core ARM Cortex A7 (ARMv7 instruction set)
- GPU: Broadcom Video Core IV @ 250 MHz; OpenGL ES 2.0 (24 GFLOPS); 1080p30 MPEG-2 and VC-1 decoder (with license); 1080p30 h.264/MPEG-4 AVC high-profile decoder and encoder
- Memory: 1 GB (shared with GPU)
- USB ports: 4
- Video input: 15-pin MIPI camera interface (CSI) connector
- Video outputs: HDMI, composite video (PAL and NTSC) via 3.5 mm jack
- Audio input: I<sup>2</sup>S

- Audio outputs: Analog via 3.5 mm jack; digital via HDMI and I<sup>2</sup>S
- Storage: MicroSD
- Network: 10/100Mbps Ethernet
- Peripherals: 17 GPIO plus specific functions, and HAT ID bus
- Power rating: 800 mA (4.0 W)
- Power source: 5 V via MicroUSB or GPIO header
- Size: 85.60mm × 56.5mm
- Weight: 45g (1.6 oz)

### 4.3. ATMEGA 328 PU IC

Atmega328PU contains 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, reset port. It is generally used in Arduino UNO board. In this project it is being used for synchronizing the raspberry pi with driver IC and Bluetooth module.

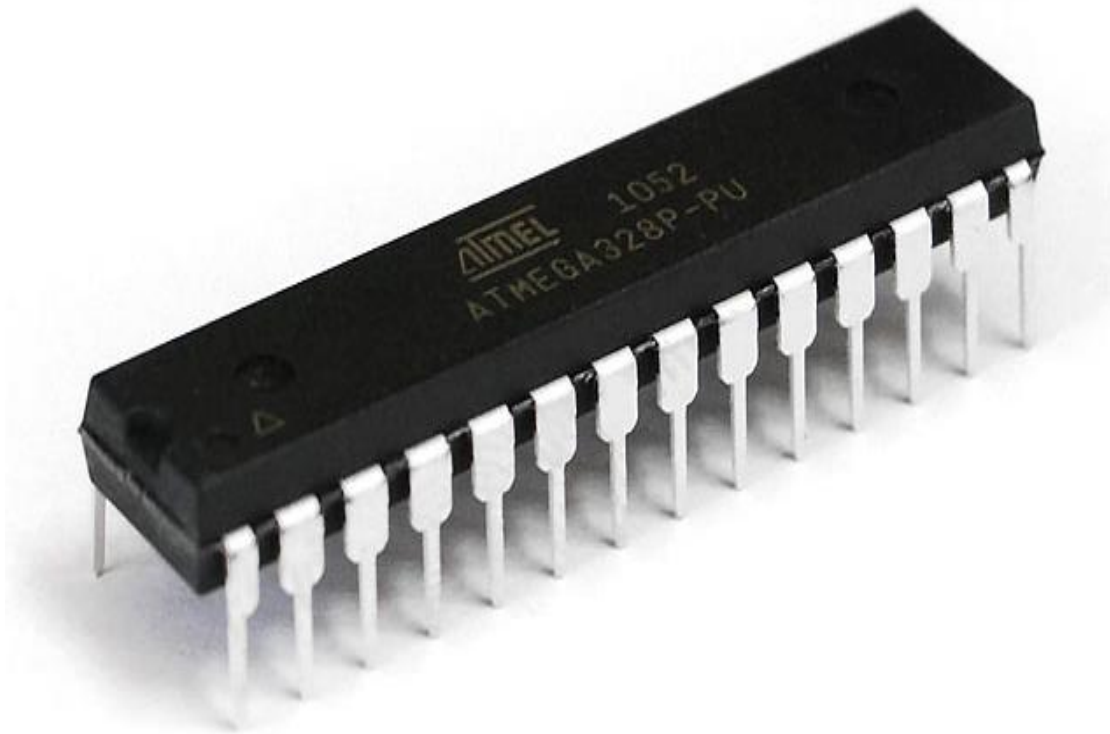


Figure 15: ATMEL ATMEGA 328P-PU

### 4.3.1. Specifications

Manufactured by Atmel, it is an 8 bit AVR RISC-based high performance microcontroller IC. It is capable of combining 32 kB In-system Programming (ISP) flash memory with read-while-write ability, it has a 1 kB Electrically Erasable Programmable ROM, a 2 kB Static RAM, 23 general purpose Input/Output (GPIO) pins, 32 general purpose registers, three timers with compare modes, serial programmable USART, a 2-wire serial interface, SPI serial port, 6-channel 10-bit Analogue to Digital (A/D) converter, a programmable timer with an internal oscillator, and five software selectable power saving modes. It operates between 1.8-5.5 volts. It achieves throughput approaching 1 MIPS per MHz.

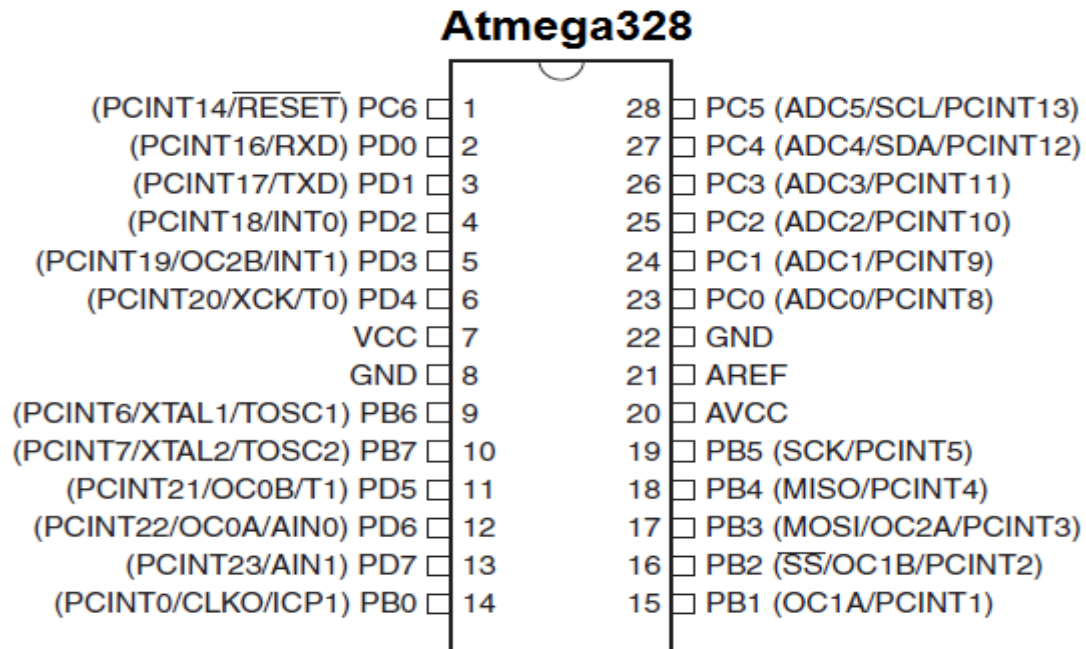


Figure 16: PIN Configuration of ATMEGA 328P

### 4.4. Bluetooth Module

HC-05 bluetooth module is a widely used option for short range wireless connectivity. It is an easily implementable Bluetooth Serial Port Protocol (SPP) module. It provides wireless connectivity over short range. The module is capable to be used in either Master or Slave configuration. This Bluetooth module is completely capable of Bluetooth V2.0+EDR (Enhanced Data Rate). It can offer 3MBPS modulation with entire 2.4 GHz radio transceiver and baseband. It takes into use CSR Blue core 04-External single chip Bluetooth system with CMOS technology and AFH.



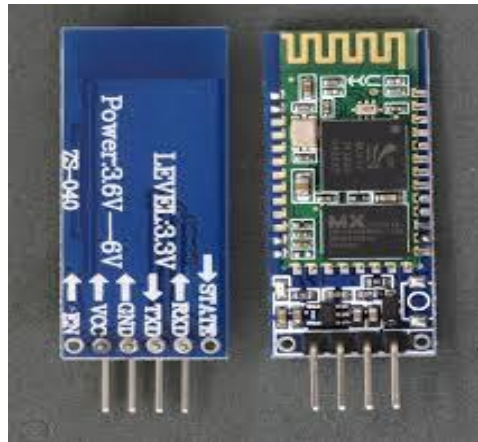


Figure 17: Bluetooth Module HC-05

## 4.5. Mechanical Pusher

### 4.5.1. Elements:

1. Disk
2. Stepper Motor
3. Motor Driver (L298n)
4. Plastic rod
5. Plastic Pusher

### 4.5.2. Construction:

A disk is used which has a movable joint connected to it. A rod is connected at the end of the joint that will remove the defective packet from the conveyor belt. The disk is connected to a stepper motor which helps the disk to move 180 degrees to and fro. Stepper motor is connected with the motor driver (L298n). The rod moves as the disk rotates 180 degrees. The analogous motion of the rod and the disk will remove the defective leaflet from the conveyer belt.

### 4.5.3. Disk:

A circular disk is employed which has diameter of 12 inches, width of 2 mm and around 150 g weight.

#### **4.5.4. Stepper motor:**

A stepper motor is a device that converts electric power into mechanical power. It is an electromechanical appliance which converts electric pulses into corresponding discrete mechanical movements. The wanted mechanical movements can be obtained by managing the pulses i.e. the pulses are provided at proper sequence as input to the motor.

##### **4.5.4.1. Specifications:**

- 5-16 Volts.
- 3 Amps current.
- 5 KG torque.
- 250 Grams weight.
- Min resolution 7.2 degrees.

##### **4.5.4.2. Application:**

The mechanical pusher will get a signal from the processor if the packet has to be removed or not. When a defective packet has been identified, the stepper motor will move forward 180 degrees to remove it. After moving 180 degrees the packet is removed and the motor moves 180 degrees backwards.

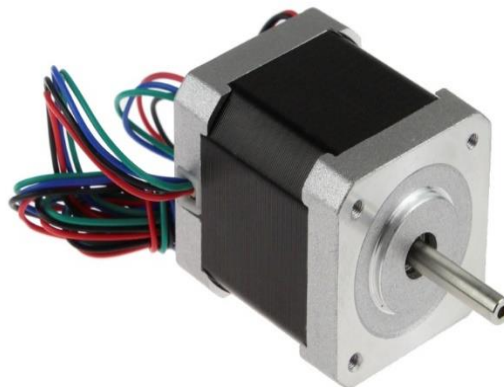


Figure 18: Stepper Motor

#### 4.5.4.3. Motor Driver (L298n):

The L298n is a device that is capable of driving inductive loads which include relays, DC motors and stepper motors by taking as input standard Transistor-Transistor Logic (TTL) levels. It is a high voltage, high current, dual full-bridge motor driver. Double H driver module uses ST L298n dual full bridge driver, an integrated monolithic circuit in a 15-lead multi watt and power SO 20 packages.

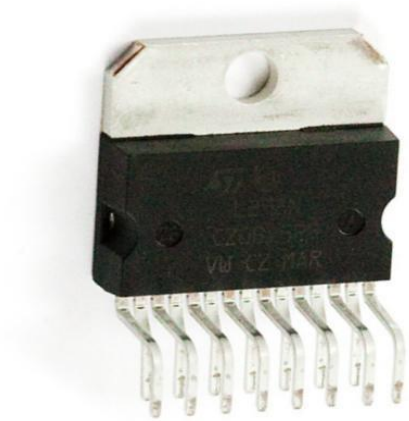


Figure 19: Motor Driver (L298n)

#### 4.5.4.4. Features:

- 46 V supply operating voltage
- 4.5-7 V DC supply voltage
- 4 Amp DC Current
- Low saturation voltage
- Stability against over temperature
- Logical '0' input up to 1.5V
- High noise immunity

#### 4.5.5. Plastic Rod and Pusher:

A plastic rod is attached to the disk, which has a length of 24 inches and a diameter of 0.2 mm. A pusher is connected at the end of the rod which removes the medicine packet from the belt.

# 5. Software Design

---

## 5.1. Android Application

We will be using android app for selecting of type of medicines we want to detect fault. Android developer software is used to make the application. In android developer software first when you will open it will ask for type of android version you want to make app for. Android app works on two things, one is JAVA language code which run in background and second is XML code which will show the developer the interface of the application being made. For every button, pop up window and window there will be a specific code running in the background of the developer software. The application in this project is named “PI MEDICINE”. It works on android version 4 above that is for all android devices having android version KitKat or above.

### 5.1.1. Explanation

Android App will be connected to Raspberry Pi processor with a bluetooth connection by the help of a bluetooth module. After successful connection process, user will tell raspberry pi processor the type of medicine to be detected. Android app will relay the command to raspberry pi and it will detect the leaflets. Application has a built in counter which will count the total leaflets passed and number of defective and acceptable leaflets. Further a limit would be set within the app for the faulty leaflets. If that limit is crossed, application will send an alert via notification of app or through sms to a number which will be given to inform in case of bulk of faulty leaflet. JAVA code for our application is divided into two portions, one is the list interface for connection of Bluetooth module with phone and second is main activity of main interface window in which every process is going on.

### 5.1.2. Interface of Application

When android application is turned on it first checks the Bluetooth. If the bluetooth is not on, the application will turn it on in the android device. After turning the bluetooth on, it displays the list of all connected devices. If the bluetooth module is not in the connected list it will not be displayed, so before turning the application on, bluetooth module has to be connected to the android device. From the display screen bluetooth module HC 05 is selected which then connects the application via the device to the hardware by the help of bluetooth module. After selecting the bluetooth module a new screen pops up on the device.

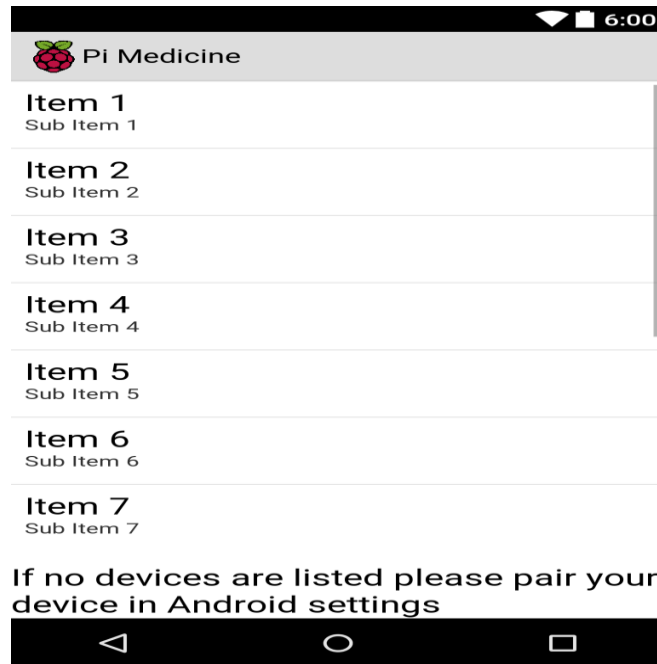


Figure 20: Interface Screen 1 of Pi Medicine

Android application will look like this after running all the codes. In interface 3 types of medicines can be seen along with a choose button to click the type of medicine we want to detect. It is also having a counter to count the passed and faulty leaflets. It also gives the options for setting the threshold limit for faulty leaflets a number which is to be notified in case of fault in bulk.

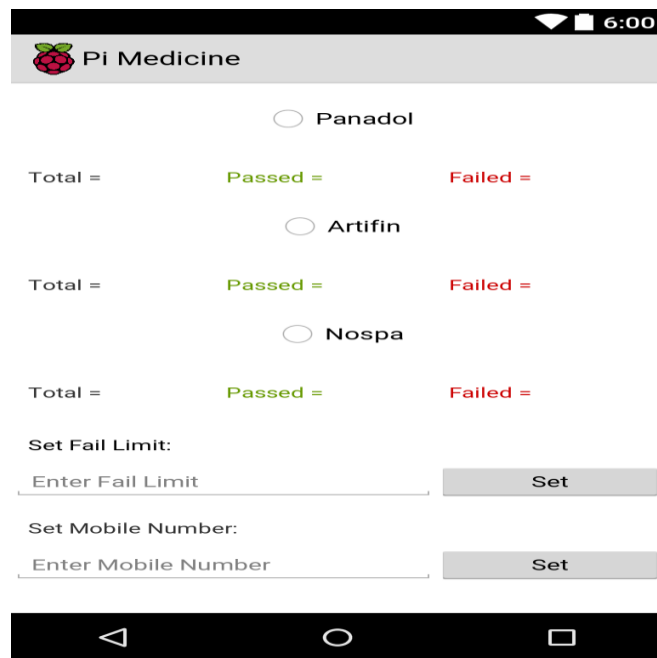


Figure 21: Interface Screen 2 of Pi Medicine

The following images show the case in which the number of defective leaflets cross the threshold and sms is being sent.

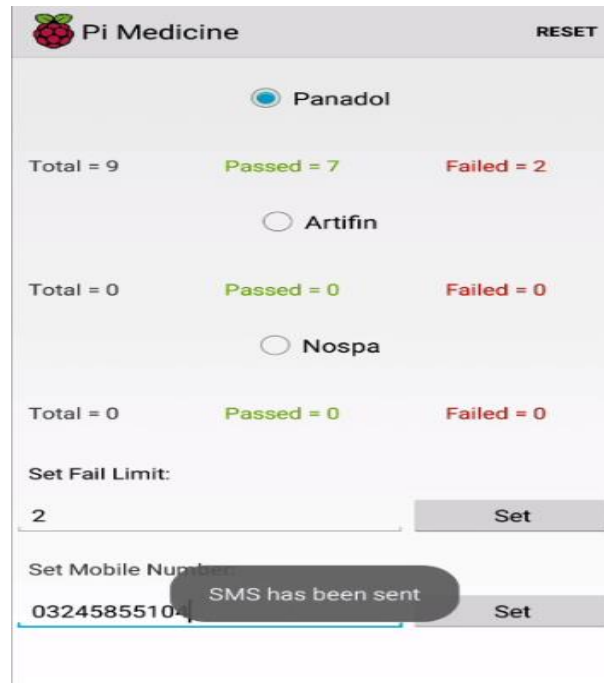


Figure 22: SMS being sent

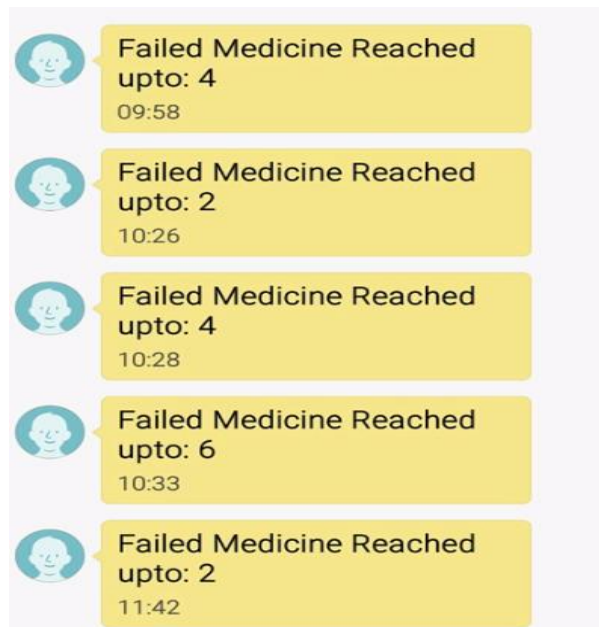


Figure 23: SMS received

## **5.2. Technique for Image Processing:**

This project is using canny edge detection technique. The Procedure of Canny edge detection technique is explained in the following steps:

1. Apply Gaussian filter to make the image smooth for noise removal.
2. Find the intensity gradients of the captured image.
3. Apply non-maximum suppression to remove bogus response to edge detection.
4. Apply double threshold to define potential edges.
5. Track edges by hysteresis: Finalize the detection of edges by eliminating all the other edges that are weak and not connected to strong edges.

## **5.3. Explanation:**

### **5.3.1. Detection of faulty leaflets:**

OpenCV is used for coding in the Python programming language. The code is quite simple. The program prompts for an input image after all the required code packages are imported and essential variables for image manipulation are created. A copy of the image is created after this. The copy is converted into greyscale (to allow for the application of suitable image processing techniques). Hough circles technique is then employed and applied on the greyscale image for circle detection. A variable 'circles' stores the result. If 'circles' is not equal to 'none', the Raspberry Pi turns on the stepper motor by motor driver which controls the movement of the mechanical pusher to remove the defective leaflet. The program also highlights the circumference of the detected circles and marks their centres with quadrilateral spots. The output image is displayed after the complete process.

### **5.3.2. Stepper Motor:**

Raspberry Pi processor signals the stepper motor when its function is called in the program. Raspberry Pi configures the GPIO pins on the stepper motor. These pins conform to the two coils attached to the stepper motor which will enable Raspberry Pi to obtain the desired movement. Two pins correspond to each of the two coils, one for positive end and the other for the negative end.

The 'forward' sub-function enables and disable the pins in a series of four actions. In each action, the pins are given a value of 0 or a 1 as per the requirement. 0 means negative end of the coil and 1 means positive end. By switching the negative and the positive ends in a specific order, the stepper motor attains a rotatory motion. The 'backwards' sub-function in the program implies the same four steps but in the opposite order which allows the control for the backward motion of the disc.

When the function is called in the program, the desired pins are set and the forward and backward sub functions are called which make the disk move 180 degrees forward for the removal of the defective packet and then 180 degrees backward for getting back.

## 5.4. Project Analysis and Evaluation

At first, a random input image was chosen for testing the code. The image contained 8 circles: one large, six medium sized and one small. Threshold was fixed such that the Raspberry Pi should not take into account any circle which was smaller in size to the medium sized circle. The result was:

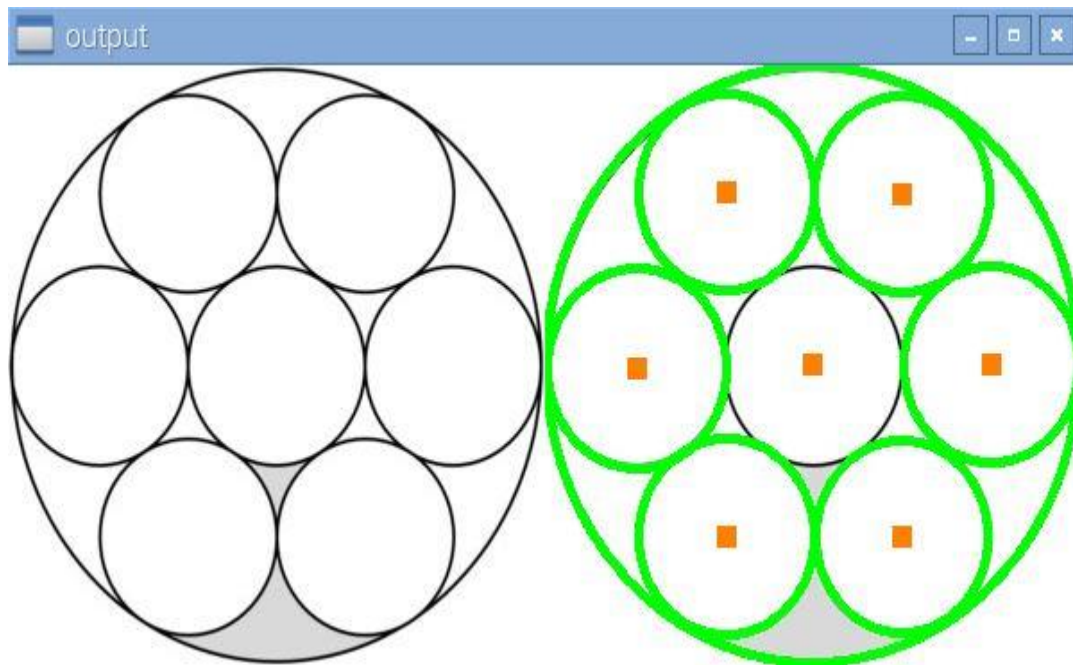


Figure 24: Simulation Result (example 1)



Selecting a suitable threshold helps to avoid the detection of false circles, in turn maximizing the efficiency (The threshold was fixed as per the size of the tablet on which the procedure is to be implemented). After effectively obtaining the objective, an actual image of a leaflet was used as the input to the program. After a few trials and variations following result was obtained:

### 5.4.1. Simulation

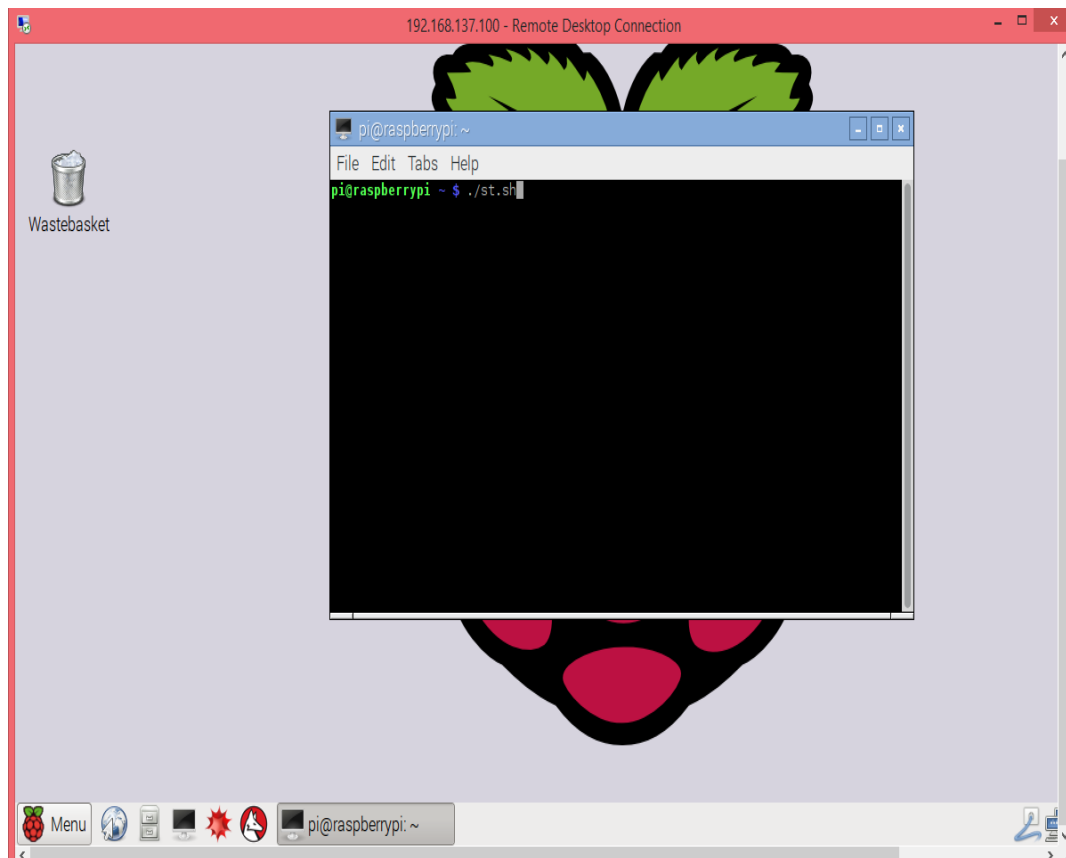
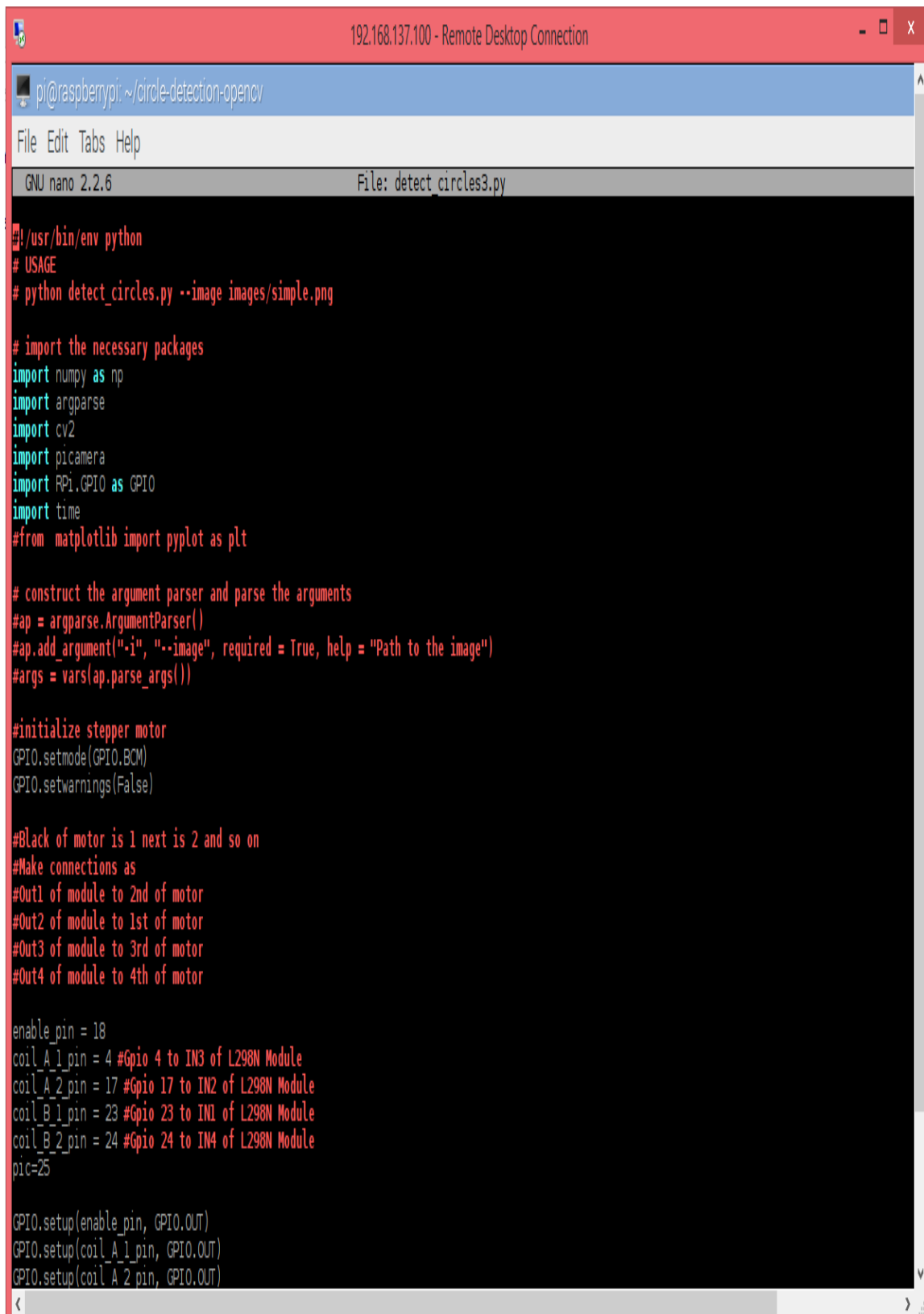


Figure 25: Function Call in Command Prompt (Raspberry Pi)

The raspberry pi processor was connected to the laptop computer using 'Remote Desktop Connection'. IP addresses were set by the connection centre which allowed the access to the processor through the computer. The output can also be demonstrated on any HDMI enabled screen by connecting the HDMI cable between Raspberry Pi and the screen.



The image shows a remote desktop connection window titled "192.168.137.100 - Remote Desktop Connection". The terminal window displays the following Python code:

```
pi@raspberrypi: ~/circle-detection-opencv
File Edit Tabs Help
GNU nano 2.2.6 File: detect_circles3.py

#!/usr/bin/env python
# USAGE
# python detect_circles.py --image images/simple.png

# import the necessary packages
import numpy as np
import argparse
import cv2
import picamera
import RPi.GPIO as GPIO
import time
#from matplotlib import pyplot as plt

# construct the argument parser and parse the arguments
#ap = argparse.ArgumentParser()
#ap.add_argument("-i", "--image", required = True, help = "Path to the image")
#args = vars(ap.parse_args())

#initialize stepper motor
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#Black of motor is 1 next is 2 and so on
#Make connections as
#Out1 of module to 2nd of motor
#Out2 of module to 1st of motor
#Out3 of module to 3rd of motor
#Out4 of module to 4th of motor

enable_pin = 18
coil_A_1_pin = 4 #Gpio 4 to IN3 of L298N Module
coil_A_2_pin = 17 #Gpio 17 to IN2 of L298N Module
coil_B_1_pin = 23 #Gpio 23 to IN1 of L298N Module
coil_B_2_pin = 24 #Gpio 24 to IN4 of L298N Module
pic=25

GPIO.setup(enable_pin, GPIO.OUT)
GPIO.setup(coil_A_1_pin, GPIO.OUT)
GPIO.setup(coil_A_2_pin, GPIO.OUT)
```

Figure 26: Python Code

The code was run on the command terminal of the interface. The circle detection segments can be seen in the image.

#### 5.4.1.1. Medicine 1:

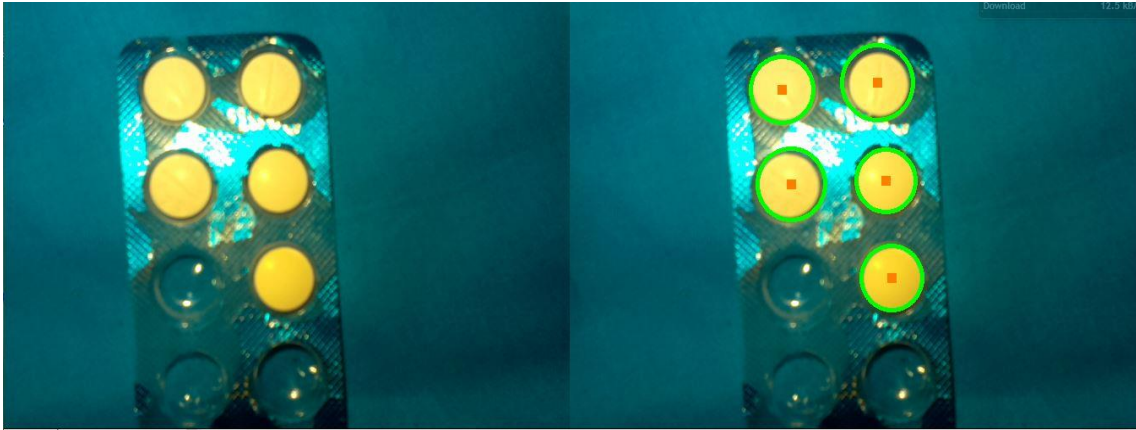


Figure 27: Simulation Result (medicine 1)

As it is clearly visible that the Raspberry Pi only detected those cavities which enclosed the tablets. The Raspberry Pi would send a signal to the mechanical pusher by the help of the stepper motor triggering it to initiate the sequence for the removal of the defective packet from the belt. The tablets used in this scenario were white in colour. This system can be implied for tablets of any colour with the same radius  $R$ .

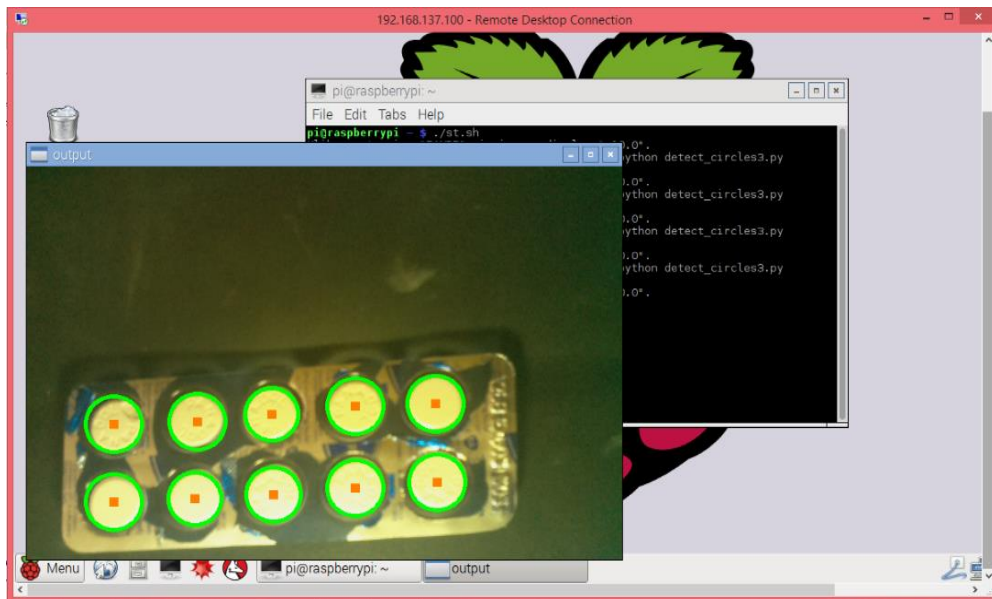


Figure 28: Output Image of faultless medicine 1 leaflet

This result depicts that all the tablets have been detected by the processor. As the packet adheres to the standard of packaging it will not be removed from the belt.

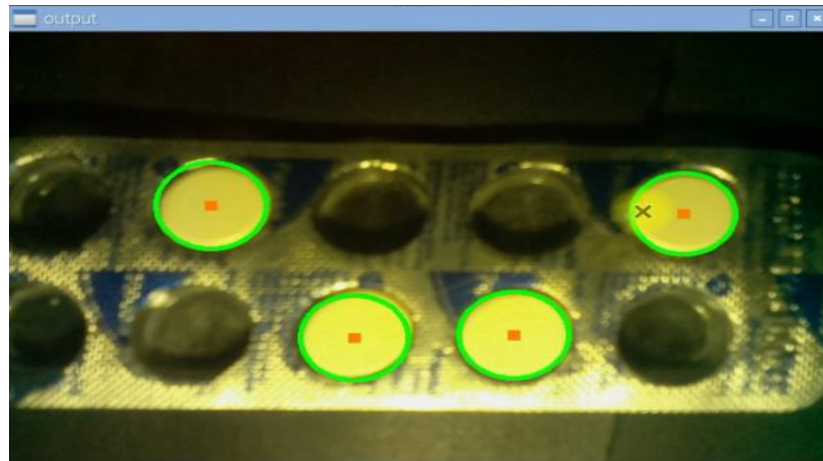


Figure 29: Output Image of defective medicine 1 leaflet

The simulation results in this scenario depict that some tablets are omitted so they have not been sensed by the camera. This leaflet does not meet the standard and will be removed from the belt by the help of mechanical pusher.

The leaflet is placed on a moving conveyor belt and as it passes over the laser sensor, a signal is sent to the processor which in turn signals the camera to take a picture. LEDs are mounted besides the camera to increase the illumination. Once the image is captured, it is displayed on the screen and a signal is sent to the raspberry pi to either remove the leaflet or let it pass.

#### **5.4.1.2. Medicine 2:**

Brown coloured Artifin with circular shape and smaller diameter as compared to medicine 1 was used as the second medicine. The following images show the real leaflet and that after going through the detection process:



Figure 30: Medicine 2

After running the image processing algorithm on this type of medicine, the following results were successfully achieved.



Figure 31: Simulation result (medicine 2)

As all the tablets are present it will be allowed to pass into the market after final packaging. If there were some defect it would be discarded by the pusher.

#### **5.4.1.3. Medicine 3:**

An image of medicinal leaflet of “NOSPA” which is in a capsule shape as input to our system. After some modifications the following results were achieved:



Figure 32: Medicine 3



Figure 33: Simulation Results (medicine 3)

The code was run on command terminal. The threshold values and capsule detection segments can be seen in the image. It is clearly visible that Raspberry Pi has successfully detected those cavities which have actual Capsules. As there are no faulty medicine as per the requirement, it would be carried by the conveyor belt to the next step of final packaging and distribution into the market. If there were any faulty leaflet the raspberry pi would send the signal to mechanical arm via stepper motor to push the faulty leaflet out of assembly line and discard it.



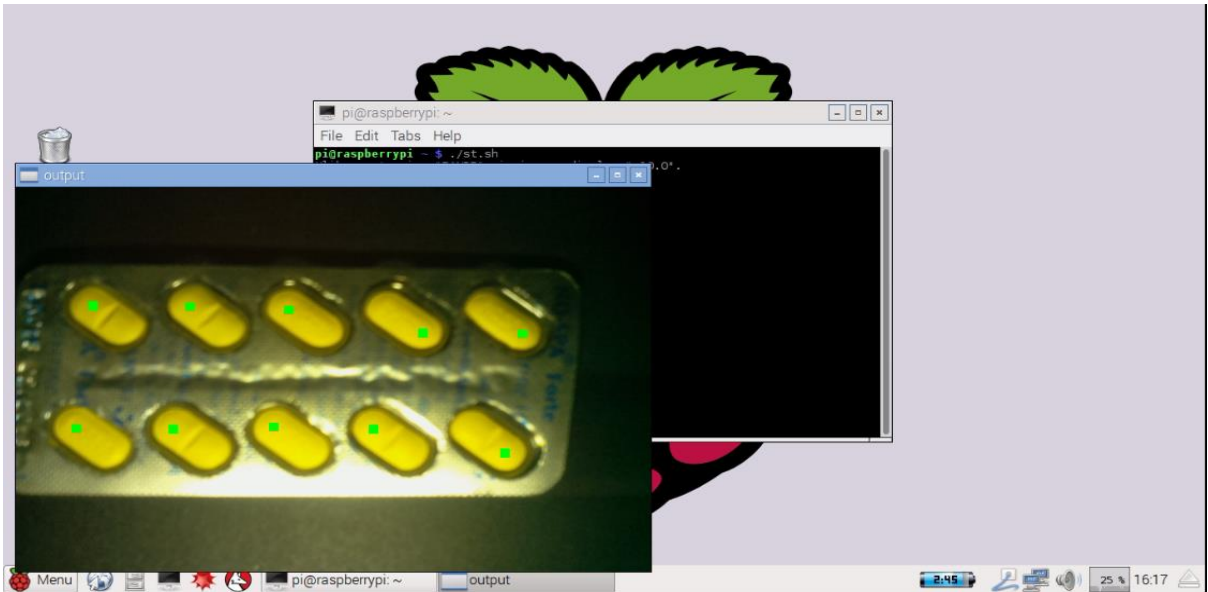


Figure 34: Output image of faultless medicine 3 leaflet

As all 10 capsules are present in the leaflet so it is allowed for final packaging and entry into the market.

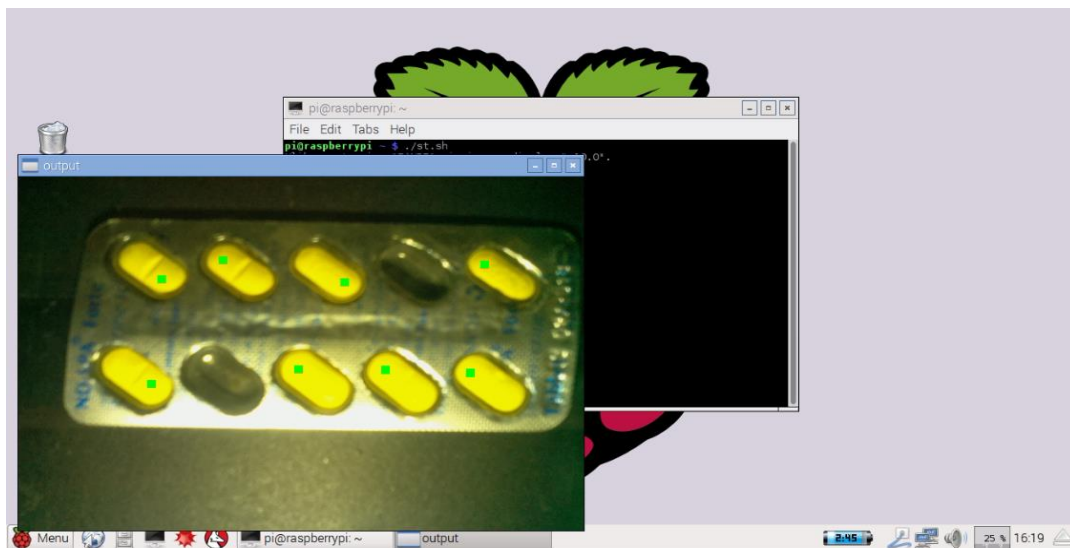


Figure 35: Output image of defective medicine 3 leaflet

This image shows that two capsules are missing and have not been detected by camera. This packaging does not meet the standard requirements and will be pushed off the conveyor belt by mechanical pusher.

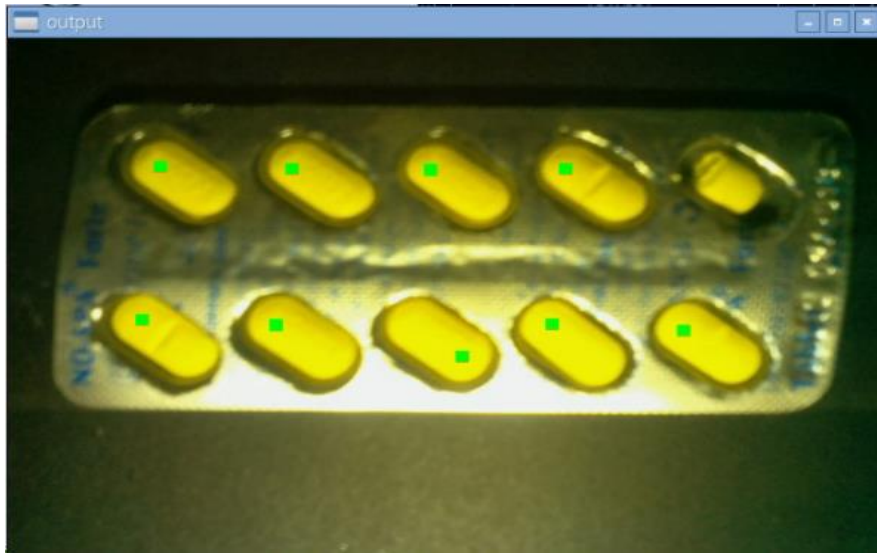


Figure 36: Leaflet containing broken capsule

This image shows that the leaflet contains the desired 10 capsules but one out of them is broken. The code also detects the broken capsule and tells the processor to signal the pusher to throw the leaflet out of the production line.



## 6. Future Work

---

This project can be revised in the future with the following enhancements.

- This project only focuses on some particular types of preset medicine tablets. In future the code can be modified to be generic in which the specifications are provided then and there with the help of application and it should be able to detect.
- This project can even be modified to detect other objects like surgical equipment and sports items etc.
- The project can make use of a more sophisticated robotic arm which could be used to pick up the object instead of pushing them out by the mechanical pusher.
- This project can make use of sensors like weight sensor etc in addition to the image processing techniques.

# 7. Conclusion

---

The pharmaceutical industry plays a very essential role in the community. It is accountable for the well-being of the public by taking into account the responsibility of resolving the health issues faced by the people. Their job needs a great deal of quality guarantee of the products.

## 7.1 Summary

The prime objective of doing this project was to contribute in helping to resolve this issue by making a Quality Control System that identifies defective tablet leaflets by the help of certain image processing techniques. This project identifies a leaflet that does not come up to the pre-set requirements and then pushes it out of the production line.

## 7.2 Objectives Accomplished

Many of the desired requirements were attained by the end of the project. It is capable of accurately identifying the defective leaflets and pushing them out of the production line by the help of the mechanical pusher.

## 7.3 Accomplishments

The project is able to accurately detect the undesired defective leaflets in the production line. After detecting the defective leaflets it discards them by the help of the mechanical pusher so that they do not enter into the market. When a defective leaflet is detected by the processor a signal is sent to the mechanical pusher to push that very leaflet out of the production line. In case there is no defective leaflet, the leaflet is passed on for final packaging.

## 8. Bibliography

---

- [1] DESIGN AND SELECTING THE PROPER CONVEYOR-BELT by Konakalla Naga Sri Ananth<sup>1</sup>, Vaitla Rakesh<sup>2</sup> AND Pothamsetty Kasi Visweswarao<sup>3</sup>  
<http://www.technicaljournalsonline.com/ijeat/VOL%20IV/IJAET%20VOL%20IV%20IS SUE%20II%20APRIL%20JUNE%202013/Vol%20IV%20Issue%20II%20Article%2012.pdf>
- [2] Development of a lemon sorting system based on color and size by M. Khojastehnazhand, M. Omid\* AND A. Tabatabaeefar.  
[http://www.academicjournals.org/article/article1380110185\\_Khojastehnazhand%20et%20al.pdf](http://www.academicjournals.org/article/article1380110185_Khojastehnazhand%20et%20al.pdf)
- [3] Automatic color object sorting system by Shubhangi Wanve<sup>1</sup>, B.G.Gawalwad<sup>2</sup>,  
<sup>1</sup>E&TC Department, SVIT Chincholi, Nasik, shubhangi16\_wanve@yahoo.co.in <sup>2</sup>E&TC Department, SVIT Chincholi, Nasik, balaji\_gawalwad@rediffmail.com  
[http://www.ijmter.com/published\\_special\\_issues/07-02-2015/automatic-color-object-sorting-system.pdf](http://www.ijmter.com/published_special_issues/07-02-2015/automatic-color-object-sorting-system.pdf)
- [4] AUTOMATIC DEFECT DETECTION AND CLASSIFICATION TECHNIQUE FROM IMAGE: A SPECIAL CASE USING CERAMIC TILES by  
G. M. Atiq ur Rahaman  
Computer Science and Engineering Discipline  
Khulna University  
Khulna 9208, Bangladesh  
atiq99@hotmail.com and  
Md. Mobarak Hossain  
Computer Science and Engineering Department  
Asian University of Bangladesh

Dhaka, Bangladesh

mobarak10jan@yahoo.com

[https://www.google.com.pk/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&sqi=2&ved=0CCAQFjAA&url=http%3A%2F%2Farxiv.org%2Fpdf%2F0906.3770&ei=t7YJVc6UAYvdUfemgJAL&usg=AFQjCNFT-X8AJZ7nFxf45CvpMbARoZGuPA&sig2=o5wVM7eg\\_6p9Fzno9j2FMA&bvm=bv.88198703,d.d24](https://www.google.com.pk/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&sqi=2&ved=0CCAQFjAA&url=http%3A%2F%2Farxiv.org%2Fpdf%2F0906.3770&ei=t7YJVc6UAYvdUfemgJAL&usg=AFQjCNFT-X8AJZ7nFxf45CvpMbARoZGuPA&sig2=o5wVM7eg_6p9Fzno9j2FMA&bvm=bv.88198703,d.d24)

[5] Surface Defects Detection for Ceramic Tiles Using Image Processing and Morphological Techniques by H. Elbehiery, A. Hefnawy, and M. Elewa

<https://www.google.com.pk/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&sqi=2&ved=0CCkQFjAB&url=http%3A%2F%2Fciteseerx.ist.psu.edu%2Fviewdoc%2Fdownload%3Fdoi%3D10.1.1.135.1354%26rep%3Drep1%26type%3Dpdf&ei=t7YJVc6UAYvdUfemgJAL&usg=AFQjCNFjFdtc29KFXxN4oowGJrD9R2-byQ&sig2=R8ZiYpn6q9BncSgnu33r4g&bvm=bv.88198703,d.d24>

[6] Application of Color Based Ore Sorting through Image Processing by Aiyush Suhasaria<sup>1</sup> and Khanindra Pathak<sup>2</sup>

<http://www.infomine.com/publications/docs/Suhasaria2012.pdf>

[7] <http://www.pyimagesearch.com>

[8] <http://www.wingodharr.wordpress.com>

# 9. Appendix A

## Automated Quality Control System

<b>Extended Title:</b> Automated Quality Control System for Detection of Faulty Medicine Packaging.
<b>Brief Description of The Project / Thesis with Salient Specifications:</b> The project will initially choose tablets/capsules from list shown on interface to detect faulty medicine leaflets. Once a leaflet, on the conveyor belt is directly below the camera an image of the leaflet will then be captured by the camera and if the leaflet is determined to be faulty a signal will be sent to the mechanical pusher to remove that particular leaflet from the conveyor belt and if more than few leaflets are faulty it will be intimated to supervisor.
<b>Scope of Work :</b> Algorithms will be used to detect faulty packaging system. Image processing will play a major role as it will be used to maintain a standard of quality at the production line.
<b>Academic Objectives :</b> Put our theoretical knowledge to practical use. Polish our skill in the field of image processing and embedded system design and communication.
<b>Application / End Goal Objectives :</b> To successfully detect a faulty leaflet, remove it from the production line, send acceptance/rejection data to the processing board and if too much rejection then inform the supervisor at back end.
<b>Previous Work Done on The Subject :</b> This technology is already implemented in other countries such as USA, UK and UAE.
<b>Material Resources Required:</b> Following are different material resources we will be using for this project:- <ul style="list-style-type: none"><li>• Raspberry pi Camera</li><li>• Raspberry Pi</li><li>• Conveyor belt</li><li>• Mechanical Pusher</li><li>• Sensors</li></ul>
<b>Special Skills Required:</b> Following are the skills required for this project:- <ul style="list-style-type: none"><li>• Signal processing.</li><li>• Image processing.</li><li>• Programming.</li><li>• Embedded system design.</li><li>• Communication.</li></ul>

# 10. Appendix B

---

## ANDROID APPLICATION CODE

### List activity code:

```
public class DeviceListActivity extends Activity {
    // Debugging for LOGCAT
    private static final String TAG = "DeviceListActivity";
    private static final boolean D = true;

    // declare button for launching website and textview for connection status
    Button tlbutton;
    TextView textView1;

    // EXTRA string to send on to mainactivity
    public static String EXTRA_DEVICE_ADDRESS = "device_address";

    // Member fields
    private BluetoothAdapter mBtAdapter;
    private ArrayAdapter<String> mPairedDevicesArrayAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.device_list);
    }

    @Override
    public void onResume()
    {
        super.onResume();
        //*****
        checkBTState();

        textView1 = (TextView) findViewById(R.id.connecting);
        textView1.setTextSize(40);
        textView1.setText(" ");

        // Initialize array adapter for paired devices
        mPairedDevicesArrayAdapter = new ArrayAdapter<String>(this,
R.layout.device_name);

        // Find and set up the ListView for paired devices
        ListView pairedListView = (ListView) findViewById(R.id.paired_devices);
        pairedListView.setAdapter(mPairedDevicesArrayAdapter);
        pairedListView.setOnItemClickListener(mDeviceClickListener);

        // Get the local Bluetooth adapter
        mBtAdapter = BluetoothAdapter.getDefaultAdapter();
    }
}
```



## Main Activity Code:

```
public class MainActivity extends Activity {

    Button btnLimitSet, btnNumSet;
    TextView txtTotal1, txtTotal2, txtTotal3, txtPassed1, txtPassed2,
    txtPassed3, txtFailed1, txtFailed2, txtFailed3;
    RadioButton btnModel, btnMode2, btnMode3;
    EditText etxtNum, etxtLimit;
    Handler bluetoothIn;
    public int total1 = 0;
    public int total2 = 0;
    public int total3 = 0;
    public int fail1 = 0;
    public int fail2 = 0;
    public int fail3 = 0;
    public int pass1 = 0;
    public int pass2 = 0;
    public int pass3 = 0;
    public int mode = 1;
    public String num = "";

    //public String msg = "Failed Medicine Reached upto: ";
    public int failLimit = 5;
    public boolean numSet = false;

    final int handlerState = 0; //used to identify handler
    message
    private BluetoothAdapter btAdapter = null;
    private BluetoothSocket btSocket = null;
    private StringBuilder recDataString = new StringBuilder();

    private ConnectedThread mConnectedThread;
    private SharedPreferences preferenceSettings;
    private SharedPreferences.Editor preferenceEditor;
    private static final int PREFERENCE_MODE_PRIVATE = 0;

    // String for MAC address
    private static String address;

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu, menu); //your file name
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(final MenuItem item) {

        switch (item.getItemId()) {
            case R.id.res:
                total1 = 0;
                total2 = 0;
                total3 = 0;
                pass1 = 0;
                pass2 = 0;
                pass3 = 0;
                fail1 = 0;
                fail2 = 0;
                fail3 = 0;
                failLimit = 5;
                num = "";
        }
    }
}
```



```

        numSet = false;
        txtTotal1.setText("Total = 0");
        txtTotal2.setText("Total = 0");
        txtTotal3.setText("Total = 0");
        txtPassed1.setText("Passed = 0");
        txtPassed2.setText("Passed = 0");
        txtPassed3.setText("Passed = 0");
        txtFailed1.setText("Failed = 0");
        txtFailed2.setText("Failed = 0");
        txtFailed3.setText("Failed = 0");
        btnMode1.setChecked(false);
        btnMode2.setChecked(false);
        btnMode3.setChecked(false);
        return true;
    default:
        return super.onOptionsItemSelected(item);
    }
}

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
    preferenceSettings = getPreferences(PREFERENCE_MODE_PRIVATE);
    total1 = preferenceSettings.getInt("Total1", 0);
    total2 = preferenceSettings.getInt("Total2", 0);
    total3 = preferenceSettings.getInt("Total3", 0);
    pass1 = preferenceSettings.getInt("Passed1", 0);
    pass2 = preferenceSettings.getInt("Passed2", 0);
    pass3 = preferenceSettings.getInt("Passed3", 0);
    fail1 = preferenceSettings.getInt("Failed1", 0);
    fail2 = preferenceSettings.getInt("Failed2", 0);
    fail3 = preferenceSettings.getInt("Failed3", 0);
    failLimit = preferenceSettings.getInt("FailLimit", 5);
    num = preferenceSettings.getString("num", "");
    numSet = preferenceSettings.getBoolean("numSet", false);
    mode = preferenceSettings.getInt("mode", 1);
    // preferenceEditor = preferenceSettings.edit();
    //Link the buttons and textViews to respective views
    btnLimitSet = (Button) findViewById(R.id.btnLimitSet);
    btnNumSet = (Button) findViewById(R.id.btnNumSet);
    btnMode1 = (RadioButton) findViewById(R.id.btnMode1);
    btnMode2 = (RadioButton) findViewById(R.id.btnMode2);
    btnMode3 = (RadioButton) findViewById(R.id.btnMode3);
    txtTotal1 = (TextView) findViewById(R.id.textTotal1);
    txtTotal2 = (TextView) findViewById(R.id.textTotal2);
    txtTotal3 = (TextView) findViewById(R.id.textTotal3);
    txtPassed1 = (TextView) findViewById(R.id.textPassed1);
    txtPassed2 = (TextView) findViewById(R.id.textPassed2);
    txtPassed3 = (TextView) findViewById(R.id.textPassed3);
    txtFailed1 = (TextView) findViewById(R.id.textFailed1);
    txtFailed2 = (TextView) findViewById(R.id.textFailed2);
    txtFailed3 = (TextView) findViewById(R.id.textFailed3);
    etxtNum = (EditText) findViewById(R.id.editTextMobile);
    etxtLimit = (EditText) findViewById(R.id.editTextLimit);

    String s = "Total = " + Integer.toString(total1);
    txtTotal1.setText(s);
    s = "Total = " + Integer.toString(total2);
    txtTotal2.setText(s);
    s = "Total = " + Integer.toString(total3);
    txtTotal3.setText(s);
    s = "Passed = " + Integer.toString(pass1);

```

```

txtPassed1.setText(s);
s = "Passed = " + Integer.toString(pass2);
txtPassed2.setText(s);
s = "Passed = " + Integer.toString(pass3);
txtPassed3.setText(s);
s = "Failed = " + Integer.toString(fail1);
txtFailed1.setText(s);
s = "Failed = " + Integer.toString(fail2);
txtFailed2.setText(s);
s = "Failed = " + Integer.toString(fail3);
txtFailed3.setText(s);
if (mode == 1) {
    btnMode1.setChecked(true);
    btnMode2.setChecked(false);
    btnMode3.setChecked(false);

    Toast.makeText(getApplicationContext(), "Panadol is Selected",
Toast.LENGTH_SHORT).show();
}
else if (mode == 2) {
    btnMode1.setChecked(false);
    btnMode2.setChecked(true);
    btnMode3.setChecked(false);
    Toast.makeText(getApplicationContext(), "Artifin is Selected",
Toast.LENGTH_SHORT).show();
}

else if (mode == 3) {
    btnMode1.setChecked(false);
    btnMode2.setChecked(false);
    btnMode3.setChecked(true);
    Toast.makeText(getApplicationContext(), "Nospa is Selected",
Toast.LENGTH_SHORT).show();
}

bluetoothIn = new Handler() {
    public void handleMessage(Message msg) {
        if (msg.what == handlerState) {
//if message is what we want
            String readMessage = (String) msg.obj;
// msg.arg1 = bytes from connect thread
            recDataString.append(readMessage);
//keep appending to string until ~
            int endOfLineIndex = recDataString.indexOf("~");
// determine the end-of-line
            if (endOfLineIndex > 0) {

if (recDataString.charAt(0) == '#')

//if it starts with # we know it is what we are looking for
            {
                String counter = recDataString.substring(1, 2);
//get sensor value from string between indices 1-5
                if (counter.equals("c")) {
                    if (mode == 1) {
                        total1++;
                        pass1 = total1 - fail1;
                        String x1 = "Total = " +
Integer.toString(total1);

                        String x2 = "Passed = " +
Integer.toString(pass1);

```

```

        txtTotal1.setText(x1);
        txtPassed1.setText(x2);
    }
    else if (mode == 2) {
        total2++;
        pass2 = total2 - fail2;
        String x1 = "Total = " +

Integer.toString(total2);

        String x2 = "Passed = " +

Integer.toString(pass2);

        txtTotal2.setText(x1);
        txtPassed2.setText(x2);
    }
    else if (mode == 3) {
        total3++;
        pass3 = total3 - fail3;
        String x1 = "Total = " +

Integer.toString(total3);

        String x2 = "Passed = " +

Integer.toString(pass3);

        txtTotal3.setText(x1);
        txtPassed3.setText(x2);
    }
}
else if (counter.equals("f")) {
    if (mode == 1) {
        fail1++;
        pass1 = total1 - fail1;
        String x1 = "Failed = " +

Integer.toString(fail1);

        String x2 = "Passed = " +

Integer.toString(pass1);

        txtFailed1.setText(x1);
        txtPassed1.setText(x2);
        if ((fail1 % failLimit == 0) && numSet) {
            String msg1 = "Failed Medicine Reached

upto: " + fail1;

            sendSMS(num, msg1);
        }
        else if (!numSet) {
            Toast.makeText(getApplicationContext(),
"Number is not set", Toast.LENGTH_LONG).show();

        }
    }
}
else if (mode == 2) {
    fail2++;
    pass2 = total2 - fail2;
    String x1 = "Failed = " +

Integer.toString(fail2);

    String x2 = "Passed = " +

Integer.toString(pass2);

    txtFailed2.setText(x1);
    txtPassed2.setText(x2);
    if ((fail2 % failLimit == 0) && numSet) {
        String msg1 = "Failed Medicine Reached

upto: " + fail2;

        sendSMS(num, msg1);
    }
    else if (!numSet) {
        Toast.makeText(getApplicationContext(),
"Number is not set", Toast.LENGTH_LONG).show();
    }
}
}

```

```

    }
}
else if (mode == 3) {
    fail3++;
    pass3 = total3 - fail3;
    String x1 = "Failed = " +
Integer.toString(fail3);
    String x2 = "Passed = " +
Integer.toString(pass3);
    txtFailed3.setText(x1);
    txtPassed3.setText(x2);
    if ((fail3 % failLimit == 0) && numSet) {
        String msg1 = "Failed Medicine Reached
upto: " + fail3;
        sendSMS(num, msg1);
    }
    else if (!numSet){
        Toast.makeText(getApplicationContext(),
"Number is not set", Toast.LENGTH_LONG).show();
    }
}
}
}
recDataString.delete(0, recDataString.length());
//clear all string data
}
}
};

btAdapter = BluetoothAdapter.getDefaultAdapter();// get Bluetooth
adapter
checkBTState();

btnNumSet.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
        num = etxtNum.getText().toString();
        if (!numSet)
            numSet = true;
    }
});
btnLimitSet.setOnClickListener(new OnClickListener() {
    public void onClick(View v) {
        failLimit = Integer.parseInt(etxtLimit.getText().toString());
    }
});

private void sendSMS(String phoneNumber, String message) {
    AlertDialog alertDialog;
    SmsManager sms = SmsManager.getDefault();
    sms.sendTextMessage(phoneNumber, null, message, null, null);
    Toast.makeText(getApplicationContext(), "SMS has been sent",
Toast.LENGTH_LONG).show();
    alertDialog = new AlertDialog.Builder(getApplicationContext())
        .setTitle("Sms Report")
        .setCancelable(true)
        .create();
    alertDialog.setMessage("SMS has been sent !");
    if(!alertDialog.isShowing())
    {
        alertDialog.show();
    }
}
}

```

```

    private BluetoothSocket createBluetoothSocket(BluetoothDevice device)
    throws IOException {

        return device.createRfcommSocketToServiceRecord(BTMODULEUUID);
        //creates secure outgoing connection with BT device using UUID
    }

    @Override
    public void onResume() {
        super.onResume();

        //Get MAC address from DeviceListActivity via intent
        Intent intent = getIntent();

        //Get the MAC address from the DeviceListActivity via EXTRA
        address =
        intent.getStringExtra(DeviceListActivity.EXTRA_DEVICE_ADDRESS);

        //create device and set the MAC address
        BluetoothDevice device = btAdapter.getRemoteDevice(address);

        try {
            btSocket = createBluetoothSocket(device);
        } catch (IOException e) {
            Toast.makeText(getBaseContext(), "Socket creation failed",
            Toast.LENGTH_LONG).show();
        }

        // Establish the Bluetooth socket connection.
        try {
            btSocket.connect();
        } catch (IOException e) {
            try {
                btSocket.close();
            } catch (IOException e2) {
                //insert code to deal with this
            }
        }
        mConnectedThread = new ConnectedThread(btSocket);
        mConnectedThread.start();
        preferenceSettings = getPreferences(PREFERENCE_MODE_PRIVATE);
        total1 = preferenceSettings.getInt("Total1", 0);
        total2 = preferenceSettings.getInt("Total2", 0);
        total3 = preferenceSettings.getInt("Total3", 0);
        pass1 = preferenceSettings.getInt("Passed1", 0);
        pass2 = preferenceSettings.getInt("Passed2", 0);
        pass3 = preferenceSettings.getInt("Passed3", 0);
        fail1 = preferenceSettings.getInt("Failed1", 0);
        fail2 = preferenceSettings.getInt("Failed2", 0);
        fail3 = preferenceSettings.getInt("Failed3", 0);
        failLimit = preferenceSettings.getInt("FailLimit", 5);
        mode = preferenceSettings.getInt("mode", 1);
        num = preferenceSettings.getString("num", "");
        numSet = preferenceSettings.getBoolean("numSet", false);
    }

    @Override
    public void onPause() {
        super.onPause();
        try {

```

```

        //Don't leave Bluetooth sockets open when leaving activity
        btSocket.close();
    } catch (IOException e2) {
        //insert code to deal with this
    }
    preferenceSettings = getPreferences(PREFERENCE_MODE_PRIVATE);
    preferenceEditor = preferenceSettings.edit();
    preferenceEditor.putInt("Total1", total1);
    preferenceEditor.putInt("Total2", total2);
    preferenceEditor.putInt("Total3", total3);
    preferenceEditor.putInt("Passed1", pass1);
    preferenceEditor.putInt("Passed2", pass2);
    preferenceEditor.putInt("Passed3", pass3);
    preferenceEditor.putInt("Failed1", fail1);
    preferenceEditor.putInt("Failed2", fail2);
    preferenceEditor.putInt("Failed3", fail3);
    preferenceEditor.putInt("FailLimit", failLimit);
    preferenceEditor.putInt("mode", mode);
    preferenceEditor.putString("num", num);
    preferenceEditor.putBoolean("numSet", numSet);
    preferenceEditor.apply();
}

//Checks that the Android device Bluetooth is available and prompts to be
turned on if off
private void checkBTState() {

    if (btAdapter == null) {
        Toast.makeText(getBaseContext(), "Device does not support
bluetooth", Toast.LENGTH_LONG).show();
    } else {
        if (!btAdapter.isEnabled()) {
            //} else {
            Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
            startActivityForResult(enableBtIntent, 1);
        }
    }
}

public void onRadioButtonClicked(View view) {
    // Is the button now checked?
    boolean checked = ((RadioButton) view).isChecked();

    // Check which radio button was clicked
    switch (view.getId()) {
        case R.id.btnMode1:
            if (checked) {
                mode = 1;
                btnMode2.setChecked(false);
                btnMode3.setChecked(false);
                mConnectedThread.write("1");
                Toast.makeText(getBaseContext(), "Panadol is selected",
Toast.LENGTH_SHORT).show();
                break;
            }
        case R.id.btnMode2:
            if (checked) {
                mode = 2;

```

```

        btnMode1.setChecked(false);
        btnMode3.setChecked(false);
        mConnectedThread.write("2");
Toast.LENGTH_SHORT).show();
        break;
    }
    case R.id.btnMode3:
        if (checked) {
            mode = 3;
            btnMode1.setChecked(false);
            btnMode2.setChecked(false);
            mConnectedThread.write("3");
            Toast.makeText(getApplicationContext(), "Nospa is selected",
Toast.LENGTH_SHORT).show();
            break;
        }
    }
}

//create new class for connect thread
private class ConnectedThread extends Thread {
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;

    //creation of the connect thread
    public ConnectedThread(BluetoothSocket socket) {
        InputStream tmpIn = null;
        OutputStream tmpOut = null;

        try {
            //Create I/O streams for connection
            tmpIn = socket.getInputStream();
            tmpOut = socket.getOutputStream();
        } catch (IOException e) {
        }

        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }

    public void run() {
        byte[] buffer = new byte[256];
        int bytes;

        // Keep looping to listen for received messages
        while (true) {
            try {
                bytes = mmInStream.read(buffer); //read bytes
from input buffer
                String readMessage = new String(buffer, 0, bytes);
                // Send the obtained bytes to the UI Activity via handler
                bluetoothIn.obtainMessage(handlerState, bytes, -1,
readMessage).sendToTarget();
            } catch (IOException e) {
                break;
            }
        }
    }
}

```

```

    }
}

//write method
public void write(String input) {
    byte[] msgBuffer = input.getBytes();

//converts entered String into bytes
    try {
        mmOutputStream.write(msgBuffer);

        //write bytes over BT connection via outstream
    } catch (IOException e) {
        //if you cannot write, close the application
        Toast.makeText(getBaseContext(), "Connection Failure",
Toast.LENGTH_LONG).show();
        finish();
    }
}
}
}
}

```



## IMAGE PROCESSING CODE

```
# import the necessary packages
import numpy as np
import argparse
import cv2
import picamera
import RPi.GPIO as GPIO
import time

#initialize stepper motor
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

#Black of motor is 1 next is 2 and so on

#Make connections as

#Out1 of module to 2nd of motor

#Out2 of module to 1st of motor

#Out3 of module to 3rd of motor

#Out4 of module to 4th of motor

coil_A_1_pin = 4      #Gpio 4 to IN3 of L298N Module
coil_A_2_pin = 17     #Gpio 17 to IN2 of L298N Module
coil_B_1_pin = 23     #Gpio 23 to IN1 of L298N Module
coil_B_2_pin = 24     #Gpio 24 to IN4 of L298N Module
pic=25                #GPIO 25 Signal from arduino to Capture
faulty = 16           #GPIO 16 Signal to Arduino about Fault.
mode_1=19             #GPIO 19 Signal from arduino to set Mode 1
mode_2 = 20           #GPIO 20 Signal from arduino to set Mode 2
mode_3 = 26           #GPIO 26 Signal from arduino to set Mode 3

GPIO.setup(coil_A_1_pin, GPIO.OUT)
GPIO.setup(coil_A_2_pin, GPIO.OUT)
GPIO.setup(coil_B_1_pin, GPIO.OUT)
GPIO.setup(coil_B_2_pin, GPIO.OUT)
GPIO.setup(pic,GPIO.IN)
GPIO.setup(faulty, GPIO.OUT)
GPIO.setup(mode_1, GPIO.IN)
GPIO.setup(mode_2, GPIO.IN)
GPIO.setup(mode_3, GPIO.IN)
GPIO.output(faulty,0)

def forward(delay, steps):
```

```

    for i in range(0, steps):

        setStep(1, 0, 1, 0)

        time.sleep(delay)
        setStep(0, 1, 1, 0)
        time.sleep(delay)
        setStep(0, 1, 0, 1)
        time.sleep(delay)
        setStep(1, 0, 0, 1)

        time.sleep(delay)

def backwards(delay, steps):

    for i in range(0, steps):

        setStep(1, 0, 0, 1)
        time.sleep(delay)
        setStep(0, 1, 0, 1)
        time.sleep(delay)
        setStep(0, 1, 1, 0)
        time.sleep(delay)
        setStep(1, 0, 1, 0)
        time.sleep(delay)

def setStep(w1, w2, w3, w4):
    GPIO.output(coil_A_1_pin, w1)
    GPIO.output(coil_A_2_pin, w2)
    GPIO.output(coil_B_1_pin, w3)
    GPIO.output(coil_B_2_pin, w4)

#capture image and save
camera=picamera.PiCamera()

#camera.resolution=(500,500)
camera.framrate=800
camera.shutter_speed=1250

#camera.exposure_mode='off'

while True:

    if (GPIO.input(pic) == True):
        camera.capture_sequence(['sdark.jpg'])

# load the image, clone it for output, and then convert it to
grayscale
    image = cv2.imread("sdark.jpg")
    output = image.copy()

```

```

        #im2=image.copy()

        #im2=cv2.GaussianBlur(image, (5,5),5)

        #cv2.addWeighted(image,3,im2,-1,0,image)
if (GPIO.input(mode_1)==True):

        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
# Convert image into binary. Only white tablets will be detected
        p,bina_img= cv2.threshold(gray, 250, 255,
cv2.THRESH_BINARY_INV) circles = cv2.HoughCircles(bina_img,
cv2.HOUGH_GRADIENT, 1.2,
100,param1=50,param2=30,minRadius=44,maxRadius=50)

# ensure at least some circles were found

        if circles is not None:

# convert the (x, y) coordinates and radius of the circles to
integers
        circles = np.round(circles[0, :]).astype("int")
                g=0

# loop over the (x, y) coordinates and radius of the circles
        for (x, y, r) in circles:

# draw the circle in the output image, then draw a rectangle

# corresponding to the center of the circle
                cv2.circle(output, (x, y), r, (0, 255, 0), 4)
cv2.rectangle(output, (x - 5, y - 5), (x + 5, y + 5), (0, 128, 255),
-1)

                g=g+1

# push if circles less than 10
                if g < 10:
                        GPIO.output(faulty,1)
                        time.sleep(1)
                        steps=18
                        delay=8

#x=raw_input("press any key")

#delay = raw_input("Delay between steps (milliseconds)?")

#steps = raw_input("How many steps forward? ")
                        forward(int(delay) / 1000.0,

int(steps))

                                time.sleep(0.2)

```

```

        #steps = raw_input("How many steps backwards? ")
        backwards(int(delay) / 1000.0,
int(steps))

        time.sleep(0.5)
        setStep(0,0,0,0)
        GPIO.output(faulty,0)
        cv2.imshow("output",output)
        cv2.waitKey(500)

#tablet 3 Nospa
elif (GPIO.input(mode_3)==True):

    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Do equilization of Image to filter Noise
    equ = cv2.equalizeHist(gray)

    # Setup SimpleBlobDetector parameters.
    params = cv2.SimpleBlobDetector_Params()

    # Change thresholds
    params.minThreshold = 10;
    params.maxThreshold = 220;

    # Filter by Area.

    # params.filterByColor = True

    # params.blobColor = 200

    # Filter by Area.
    params.filterByArea = True
    params.minArea = 1600

    # Filter by Circularity
    params.filterByCircularity = True
    params.minCircularity = 0.01

    # Filter by Convexity
    params.filterByConvexity = True
    params.minConvexity = 0.8

    # Filter by Inertia
    params.filterByInertia = True
    params.minInertiaRatio = 0.1
    detector = cv2.SimpleBlobDetector_create(params)

# Detect blobs.
keypoints = detector.detect(bina_img)

im_with_keypoints = cv2.drawKeypoints(bina_img, keypoints,
np.array([], (0,0,255)), cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)

```

```

# ensure at least some circles were found
    if im_with_keypoints is not None:

# convert the (x, y) coordinates and radius of the circles to
integers    im_with_keypoints = np.round(circles[0,
:]).astype("int")
            g=0

# loop over the (x, y) coordinates and radius of the circles
    for (x, y, r) in circles:

# draw the circle in the output image, then draw a rectangle

# corresponding to the center of the circle
    cv2.circle(output, (x, y), r, (0, 255, 0), 4)
    cv2.rectangle(output, (x - 5, y - 5), (x + 5, y + 5), (0, 255,
0), -1)

            g=g+1

# push if circles less than 10
    if g < 10:
        GPIO.output(faulty,1)
        time.sleep(1)
        steps=18
        delay=8
        #x=raw_input("press any key")

#delay = raw_input("Delay between steps (milliseconds)?")

#steps = raw_input("How many steps forward? ")
        forward(int(delay) / 1000.0,
int(steps))
        time.sleep(0.2)
#steps = raw_input("How many steps backwards? ")
        backwards(int(delay) / 1000.0,
int(steps))
        time.sleep(0.5)
        setStep(0,0,0,0)
        GPIO.output(faulty,0)
        cv2.imshow("output", output)
        cv2.waitKey(500)

```

## STEPPER MOTOR CODE

```
void setup() {  
  
    pinMode(9,OUTPUT);  
    pinMode(10,INPUT);  
    pinMode(6,OUTPUT);  
    pinMode(7,OUTPUT);  
    pinMode(8,OUTPUT);  
  
    digitalWrite(6,HIGH);  
    digitalWrite(7,LOW);  
    digitalWrite(8,LOW);  
    Serial.begin(9600);  
  
}  
  
int x=0;  
char t='1';  
void loop()  
{  
if (Serial.available(>0)  
  
{  
    t=Serial.read();  
    if (t=='1')  
  
{  
        //Serial.println("1 is selected");
```

```

    digitalWrite(6,HIGH);
    digitalWrite(7,LOW);
    digitalWrite(8,LOW);
}

else if (t=='2')
{
    //Serial.println("2 is selected");

    digitalWrite(6,LOW);
    digitalWrite(7,HIGH);
    digitalWrite(8,LOW);
}

else if (t=='3')
{
    //Serial.println("3 is selected");
    digitalWrite(6,LOW);
    digitalWrite(7,LOW);
    digitalWrite(8,HIGH);
}
}

if (digitalRead(10)==HIGH)
{
    Serial.print("#");
    Serial.print("f");
    Serial.print("~");
    while(digitalRead(10));
}

```

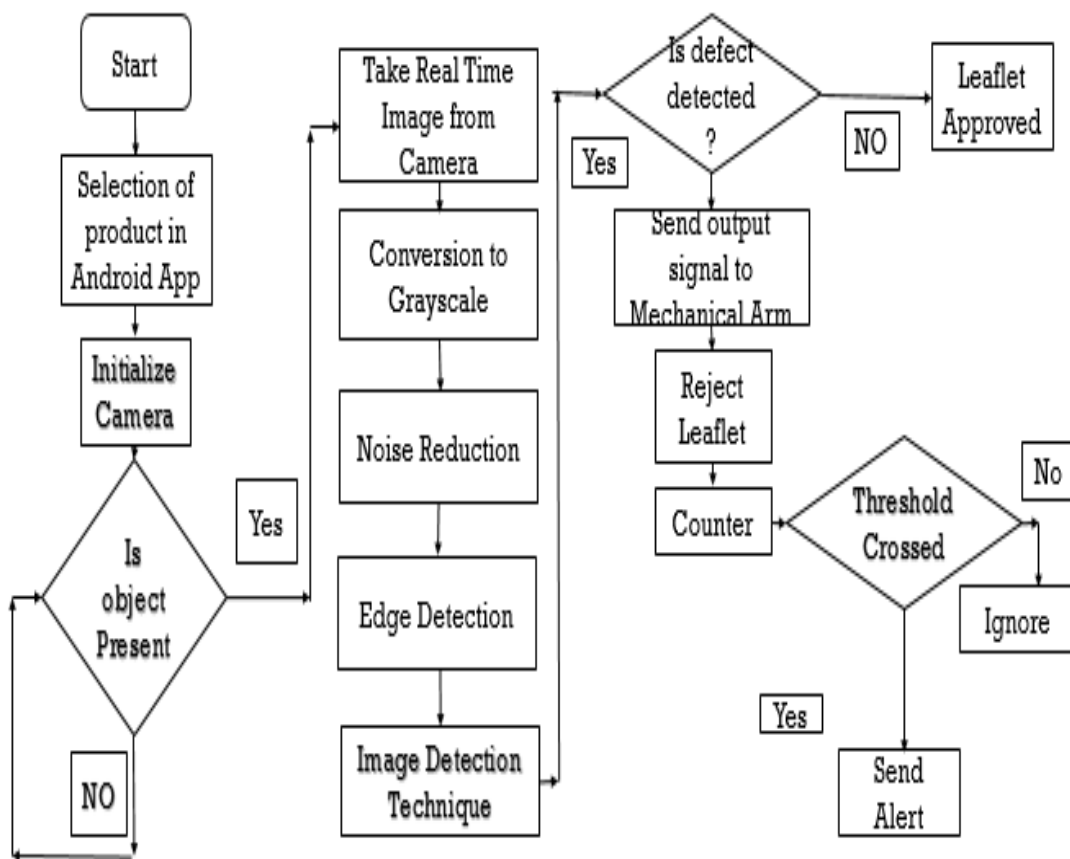
```
}

x=analogRead(A3);
if (x<350)
{
  digitalWrite(9,HIGH);
  Serial.print("#");
  Serial.print("c");
  Serial.print("~");
  delay(50);
  digitalWrite(9,LOW);
  while(analogRead(A3)<350);
}
// Serial.println(x);
delay(10);
}
```



# 11. Appendix C

## DETAILED FLOW CHART



## 12. Appendix D

---

<b>Items</b>	<b>Cost</b>
<b>Raspberry Pi</b>	5,500
<b>Raspberry Pi Camera</b>	3,500
<b>Digital Multi Meter</b>	1500
<b>Soldering Iron</b>	250
<b>Soldering wire</b>	200
<b>Circuit Components</b>	4,500
<b>Stepper Motor and Mechanical Pusher</b>	2,000
<b>AC Adapters and Driving Motors</b>	2,000
<b>Lights</b>	100
<b>Total</b>	<b>19,550</b>