Low Cost Object Sorting Robotic Arm using Raspberry Pi

Viren Pereira[1], Vandyk Amsdem Fernandes [2] and Junieta Sequeira [3]
[1]Dept. of General Engineering
[2, 3] Dept. of Electronics & Telecommunication Engineering
[1, 2, 3] Shree Rayeshwar Institute of Engineering & Information Technology,
Shiroda - Goa - India
[1]virenbusiness@gmail.com, [2]vandyk1115@gmail.com & [3]junieta2206@gmail.com

***Abstract* — Usually sorting of objects is carried out manually using human labor. Recognizing a particular object and placing it in the required position is a tiring work especially in the field of industry where in one has to sort a bulk of objects in quick time and also the weight is greater than what a human can carry. This is when automation plays a major role. In this paper we are considering all these factors along with the cost to make the process more efficient. We use Raspberry Pi, which is an open source Linux based board. Raspberry Pi has found its way in major in number of useful & versatile applications in robotic systems. But never the less this system is new & hence Latest Technology takes time to be uncovered. Therefore, not much articles are available, hence our goal will be to investigate its applicative and cost effective use as a robotic arm in Object Sorting. Thereby Revolutionizing Robotic Systems in Industrial manufacturing plants by making them cheaper, compact & along with the same reliability as that of a dedicated PC.**

Furthermore we make use of a camera module which captures the image of the object. This image is processed using GNU Octave to determine the color and the shape of the object. GNU Octave is an open source language similar to MATLAB and hence making it portable. All the processed information in Octave is than relayed to a microcontroller which in turn controls the movement of the robotic arm which will segregate the objects into its respective compartments. Here we sort objects of three different shapes and colors, i.e. Square, Circle and Triangle (As seen from the Top-View of a Cube, Sphere/Cylinder and Triangular-Prism aligned vertically) and RGB colors i.e. Red, Blue and Green respectively.

Keywords-Object sorting; Raspberry Pi; Octave; Robotic arm; Open source; Automation; Arduino

I. INTRODUCTION

In this fast growing industrial age every industrial unit needs speed in manufacturing. Robotics has found a wide application in industries. Automation provides far better service to customers eliminating the monotonous work by human, achieving accuracy and speed in work. They are high in demand and are used to carry out most of the work which saves time and is more efficient. The use of Open-Source environment makes it cost effective, Linux based Operating system used in Raspberry Pi (Raspbian OS), and GNU Octave, Python and Arduino IDE are freely available for user to use and also to develop [2]. The paper is written with a holistic approach keeping the application in the field of industry to provide an automated material handling solution. The purpose of this is to provide a robotic arm that sorts the objects in the most efficient way and in also cost effective manner.

Technology that is expanding and is widely used at present is image processing. Image processing is basically a form of signal processing, where the input is an image which is processed to obtain some values, parameters or the set of characteristics related to that particular image.

II. PROJECT DESCRIPTIION

A. *Raspberry Pi and Arduino*

Lots of components are available in the market that is used for carrying out image processing. The use of Raspberry Pi in our project has made it more cost effective solution. Raspberry Pi is a small device, is an open source and is flexible platform for experimentation and fun. Since it is an open source, changes can be made to it as and when required. It is a Linux based board. One can install various different free software for vivid purposes.

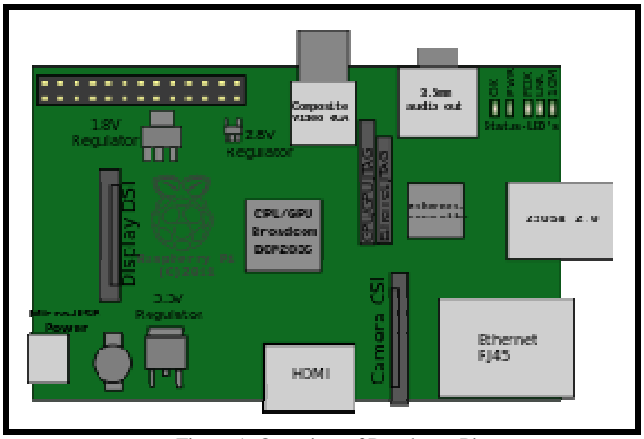

Figure 1. Overview of Raspberry Pi

The board has Broadcom BCM2835 on chip which includes an ARM1176JZF-S. This chip is 32 bit, 700 MHz System on a Chip which is built on the ARM11 architecture. It has Video Core IV GPU, and was originally shipped with 256 megabytes of RAM, later upgraded to 512 MB [1]. It does not have storage drive but uses SD card for booting and

long term storage. The Raspberry Pi runs Raspbian OS and is programmed using GNU Octave version 3.6.4 and Python 2.7.6, which is an open source [2].

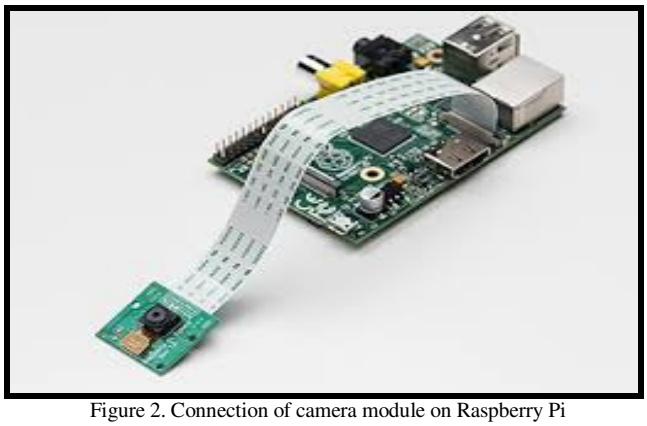

Figure 2. Connection of camera module on Raspberry Pi

The presence of the object is sensed by using an IR sensor. The signal obtained from the sensor is given to Raspberry Pi that initializes the process of image processing. Here camera is used to obtain the image of the object. This camera is directly connected to the pi at camera CSI connector provided on the board.

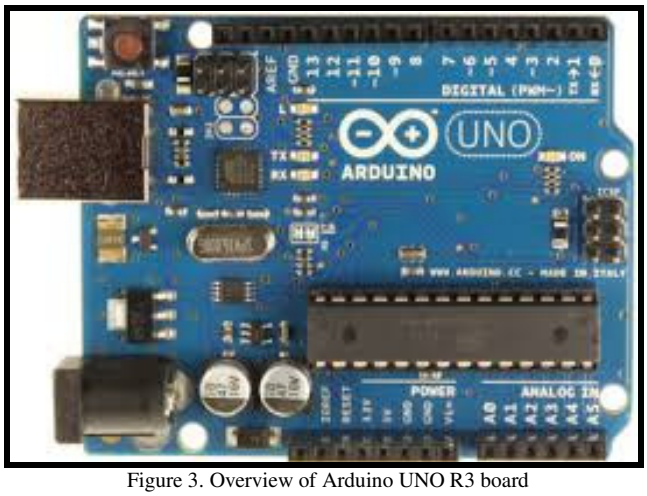

Figure 3. Overview of Arduino UNO R3 board

The initialization of the camera is done by following the below mentioned steps:

Install the Raspberry pi camera by inserting the cable into the Raspberry pi. The cable slots into the connector situated between the Ethernet and HDMI ports, with the silver connector facing the HDMI ports.

Enable the camera ⇒ Finish ⇒ Reboot Raspberry pi

Raspistill- o image.jpg

This command is to capture the image in raspberry pi. Once tested in the command line the following code will capture an image and store it successfully.

An Arduino board is used to interface Raspberry Pi and the servo motor control board. Serial communication is used to send serial messages from Raspberry Pi to controller to move the Arm accordingly. There is one serial communication port available on Arduino. The other ports are implemented using software serial communication from which more ports for data transfer are obtained.

The Robotic arm is having 4 degree of freedom joints and a gripper. It is actuated using 6 standard size servos out of which 4 are normal Nylon gears standard and 2 are Metal Gear High Torque servo motor. Servo motors are used for the movement of the arm in the desired direction. Servos are controlled by using pulse width modulation (PWM), through the control wire. Servo motors we used can usually only turn 90 degrees in either direction for a total of 180 degree. For higher application full-360 rotating servos can be used. The PWM sent to the motor determines position of the shaft, and based on the duration of the pulse the rotor will turn to the desired position. These motors are connected to the servo controller board. Which has Atmega8 and this receives data serially and controls the arm accordingly. The power required for the Robotic Arm is 5V-7.2V, 8 Amps. This can be given using a suitable adapter or using a SMPS.

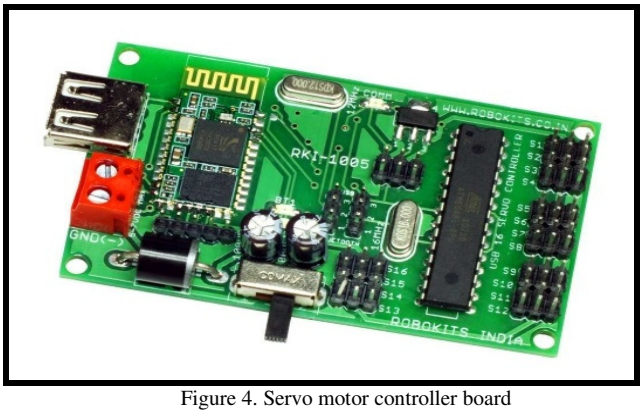

Figure 4. Servo motor controller board

B. Microcontroller

The microcontroller Atmega328 which is on the Arduino board is a high-performance Atmel 8-bit AVR RISC-based microcontroller. It combines 32KB ISP flash memory with read-while-write capabilities, 1KB EEPROM, 2KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, 6-channel 10-bit A/D converter, programmable watchdog timer with internal oscillator, and five software selectable power saving modes [4]. The device operates between 1.8-5.5 volts.

C. Image Processing

There are many Image Processing software's that are available today. The companies in today's world look out for cost effective methods to do the task. Open Source software's provide us with that solution. It helps to reduce the cost drastically as they are freely available. Also the

reason for choosing GNU Octave is because it is quite similar to MATLAB and hence provides portability. Also Open Source provides further applications and development of the software. Octave is still in its developmental phase. As for now it may not be powerful as MATLAB but in future it can well be developed as MATLAB equivalent open source solution. But it does provide the requirement for our task and we will try to exploit this Open source solution in our paper.

For Image processing first the image is captured through Pi-camera and saved as image.jpg with resolution of 600x400 (The purpose of lower resolution being the speed of processing at the same time we would like to get a quality image). The object has to be focused centrally with respect to the camera. Solution to use Flash light if required when the light is dark is also provided to get a good image for processing. The objects selected are uniform in color and shape. This are the conditions we choose for image processing.

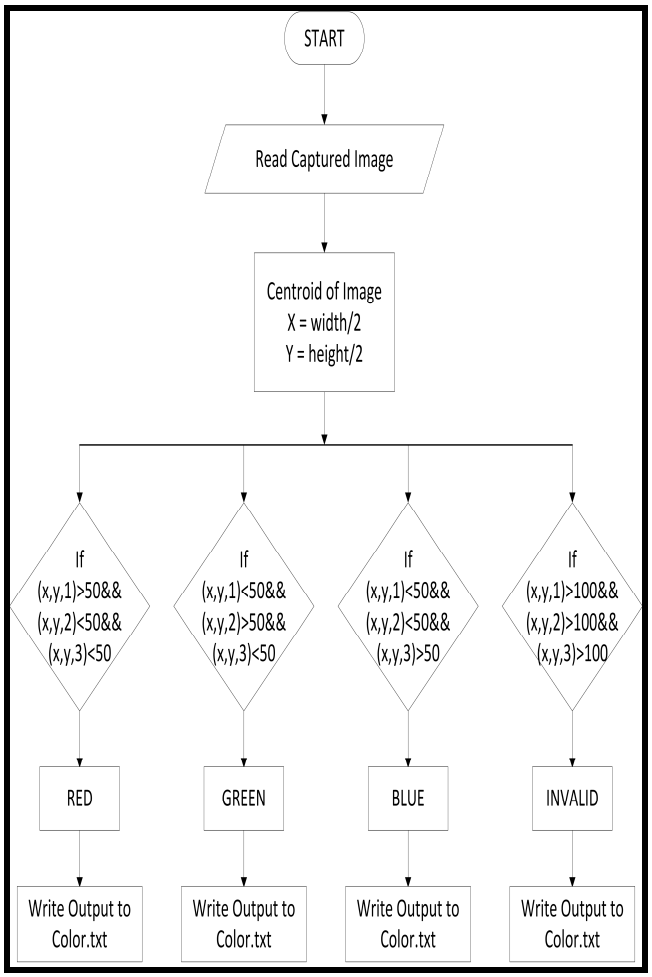

Figure 5. Flowchart for Color Detection

Algorithm for Color Detection:

Step 1: Read the image captured by the Camera.

Step 2: Take the Centroid of the image that is captured to get the center pixel (choosing of a single pixel reduces processing time, Centroid selected as the object is aligned centrally with respect to camera thus it will provide better pixel quality at the centroid). Store its pixel value.

Step 3: Check the intensity of RGB in the centroid pixel and accordingly classify object as Red, Blue and Green. Example: (x,y,1) which determines Red pixel if it is greater than 100 in 0-255 range of a 8 bit image and intensity of Green and Blue ((x,y,2) for Green and (x,y,3) for Blue) are less than 100 than we classify the object to be Red.

Step 4: Open a text file Color.txt

Step 5: Save the output color in the text file for further use. (The output is written to the file so that it can be used later with the output of shape detection to classify the object in the two parameters and there after this output can be relayed to the microcontroller for controlling of the Robotic Arm).

Step 6: Close the file.

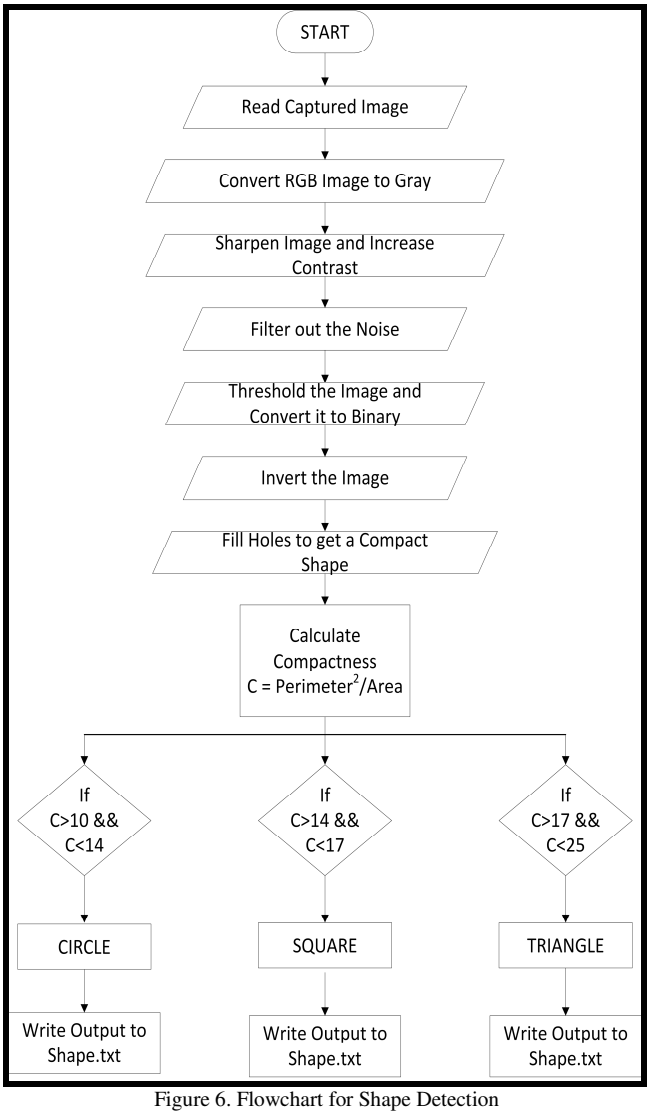

Figure 6. Flowchart for Shape Detection

Algorithm for Shape Detection:

Step 1: Read the image captured by the Camera.

Step 2: Convert the RGB image into Grayscale image.

Step 3: Sharpen the image and increase the contrast of the image to a desired level.
Step 4: Filter out the noise from the image.
Step 5: Threshold the image and convert it into a binary image.
Step 6: Invert the image. (Inverting is done cause of usage of function region props which calculates the white pixels. Inverting the image inverts the objects pixel from black to white, which enables the calculation of area and perimeter of the white pixels which resembles the object).
Step 7: Insert white pixels if there are holes found within the object mass pixels. This makes a compact representation of object pixels removing the minor holes in the image of the object.
Step 8: Using region props calculate Area and Perimeter of the object.
Step 9: Calculate compactness of the object c = Perimeter^2/Area. So if c>10 and c<14 will mean it's a circle/cylinder, c>14 and c<17 will mean it is a square/cube and c>17 and c>25 will mean it's a triangle (Equilateral). Value of c is mathematically calculated and proved to match the output. The range is provided so as to give more accuracy and reduce error due to noise and other factors in the image.
Step 10: Open a new text file.
Step 11: Save the output to the text file.
Step 12: Close the file.

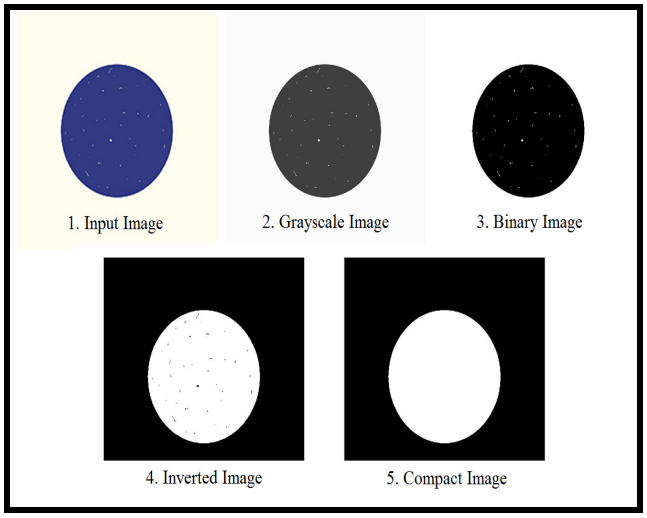

Figure 7. Image processing steps involved to get a compact image of the object

The figure 7 shows the processing of an input image to a compact figure. Here the input image is already sharpened and the contrast is increased using the camera features. In subsequent steps the image is being transformed to more compact structure through which we can easily calculate the number of white pixels to calculate area and also the perimeter of the object.

III. WORKING

The block diagram shown in Figure 5 consists of Raspberry pi, Raspberry pi camera module, Arduino, servo controller and robotic arm. Initially the IR sensor detects whether the object is present or not. If it senses the presence of the object than it sets raspberry pi camera ON and captures the image of the object. This captured image is saved on to the raspberry pi. The captured image is then processed using Octave which recognizes the color and shape of the object. The output of which is written and saved on to a text file. The Raspberry pi further sends the message to Arduino using Python script which reads the output from the two text files and sends serial message to Arduino which is used to interface the Raspberry pi and servo motors via the servo controller with the robotic arm.

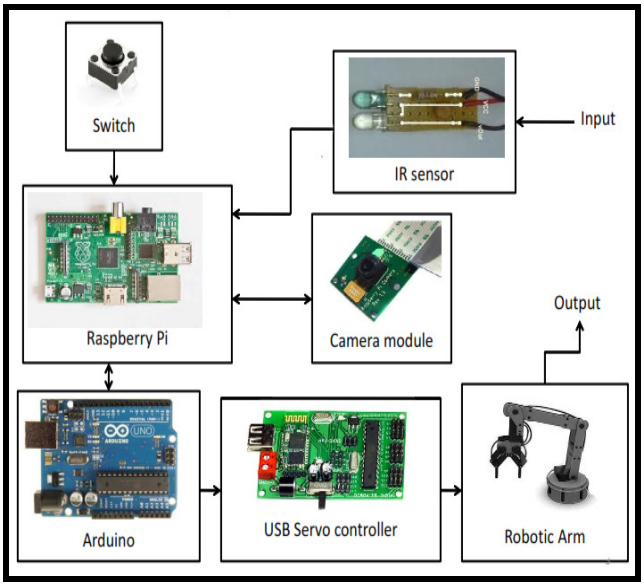

Figure 8. Block Diagram

Arduino receives the incoming message or characters from the Raspberry Pi and accordingly it controls the motion of the arm, i.e. it guides the Arm to pick the object and place it in the correct box. The motion of the arm to respective boxes is predefined within the program which is written on to the Arduino board. Motor controller board is connected to the Arduino board and the Arduino also communicates to the controller via serial communication. These instructions are given by the Arduino to the controller accordingly the arm picks the object and sort the object based on its color and shape in the appropriate box. Thus reducing human labor of sorting and arranging the objects based on the color in more efficient manner and in much lesser time.

As shown in Figure 5, the working model of the object sensing robotic arm was done, & sorted similar objects without any flaws. Hence, making it a low cost system, which can find its way in Industrial robotics systems bearing variety of PNP (Pick & Place) operations. Thereby, making this system in industries very affordable, along with the same reliability as that of a dedicated PC.

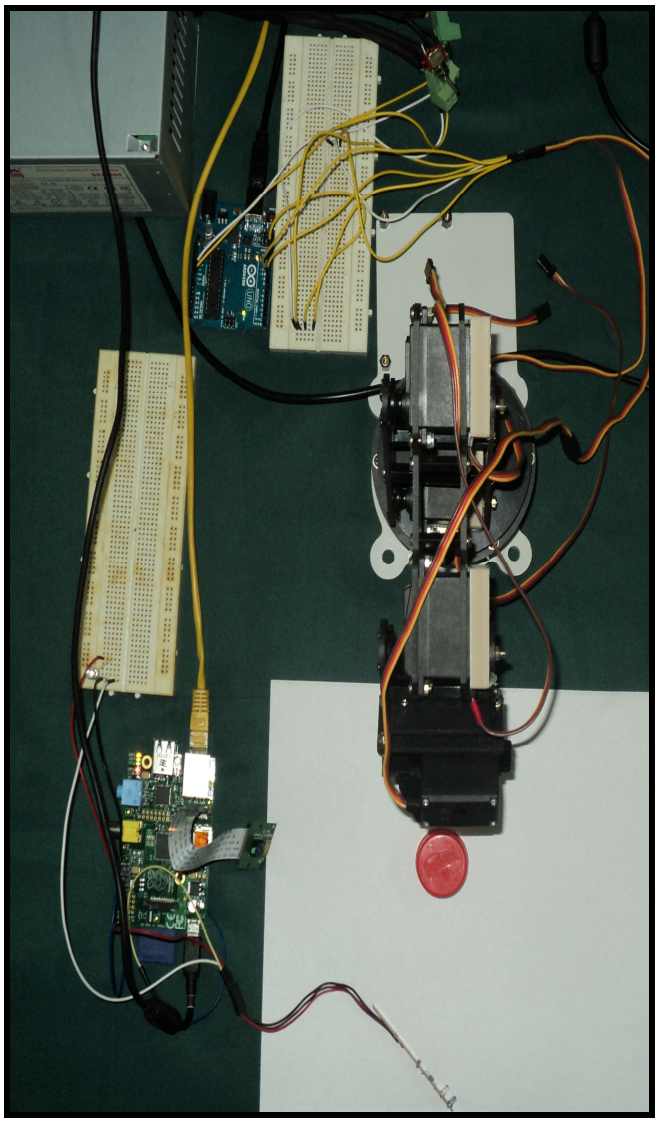

Figure 9. Setup of the Raspberry Pi, Arduino and Robotic Arm

IV. Application

This project finds its application in industry in pick and place operations. It provides a low cost system with good accuracy in sorting objects of different color, size and shapes. This can be used in many factories; one good example can be a pencil factory. It can be used for sorting of ripe and unripe fruits and other agricultural products. It can be used in cranes which require sorting of cargo containers based on colors. Simultaneously it provides a remote access to the system so monitoring and debugging of errors becomes easier over the network.

V. Future Implementation

It can be used in industrial application to provide speedy, low cost pick and place systems, with advent of raspberry pi which in future will have models with much improved RAM thus providing a low cost solution, fruit harvesting can be done depending on color of ripe and unripe fruits by making it mobile Robotic Arm and further application can even help to detect disease or abnormalities in fruits, and can be used in other researches concerning textures and shapes. The arm is light in weight because of the use of servo motors and also provides great accuracy of handling. We can define many colors using just one code instead of getting different codes as done while using a color sensor, thus making it a robust solution. Further applications can provide an android application or website through which the full system could be controlled via internet from anywhere in the world. The system could be Started/Stopped via the Android application, simultaneously providing the live video feed of the ongoing on site process and also provide the data of number of objects sorted and the count of objects sorted out into the various categories.

The speed of processing can be further increased on by using much light weight operating systems such as Arch Linux which consumes less memory.

VI. Results & Discussion

After implementing the image processing algorithms, the color detection and shape detection algorithm gave accurate results almost every time. However dark shadow of the object around it can lead to wrong measurements in case of shape detection which was overcome by adoption of flash light placed exactly above the object next to the camera and would glow for about 3 seconds while taking the photo of the object. This helped to get a clear picture of the objects, eliminating shadows and thereby achieving good accuracy in object color and shape detection.

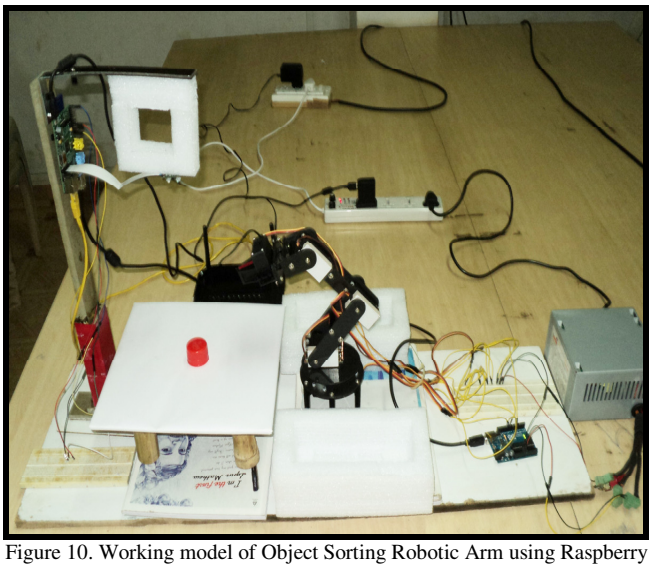

Figure 10. Working model of Object Sorting Robotic Arm using Raspberry Pi

The time taken for object detection is 2-3 seconds so the system can sort at least 20-30 objects per minute and classify them with respect to their color and shape. Color sorting algorithm on its own takes hardly a second to

identify color and its accuracy is 100% in normal room light as well as with a flash light. Shape detection however gave about 85% accuracy in normal room light which however with use of flash light it gave about 99% accuracy. Overall in a test of 200 objects of different shapes and sizes 197 objects were sorted correctly with accuracy of almost 99% and at the rate of 24 objects per minute. The number of objects sorted can also vary with respect to the robotic arm routines, however higher the speed of the arm more is the wearing of servo motors which should be taken care of.

Higher the resolution of the image better is the result obtained but it reduces the speed of processing hence a suitable resolution is to be maintained, resolution of 200 to 600dpi gives the desired results with the raspberry pi camera module.

The main intension of this project is to provide a robotic arm that will sort the objects by detecting its shape and color in the appropriate boxes which can be implemented in industrial sorting and pick and place applications, thus providing a cost efficient system. Hence this does away with a dedicated PC for such systems, which can be implemented using Raspberry Pi. Thereby saving cost, & making robotic drive systems simpler, reliable, cheaper & easier to build.

It also provides a replacement for MATLAB i.e. GNU Octave. There are other options like Open CV but the sole intention of the project is to provide a portable solution to MATLAB at much lower cost. As in we concluded that Octave and Python look like promising tools that may provide an alternative to MATLAB without compromising performance and productivity [5].

REFERENCES

[1] Getting started with Raspberry Pi by Matt Richardson & Shawn Wallace, Published by O'Reilly Media, Inc; December 2012, Page 2.

[2] "Raspbian."[Online].Available:http://www.raspbian.org/downloads

[3] "Raspberrypi."[Online].Available: http://www.raspberrypi.org

[4] "Arduino." [Online]. Available: http://www.arduino.cc/

[5] Chaves, J.C.; Nehrbass, J.; Guilfoos, B.; Gardiner, J.; Ahalt, S.; Krishnamurthy, A.; Unpingco, J.; Chalker, A.; Warnock, A.; Samsi, S., "Octave and Python: High-Level Scripting Languages Productivity and Performance Evaluation," HPCMP Users Group Conference, 2006 , vol., no., pp.429,434, 26-29 June 2006.

[6] Szabo, R.; Lie, I., "Automated colored object sorting application for robotic arms," Electronics and Telecommunications (ISETC), 2012 10th International Symposium on , vol., no., pp.95,98, 15-16 Nov. 2012.

[7] Edwards, C., "Not-so-humble raspberry pi gets big ideas," Engineering & Technology , vol.8, no.3, pp.30,33, April 2013.

BIOGRAPHIES

First 1. Author was born in Margao, Goa, India, in 1985. He received the Diploma in Electrical Engineering from the Goa Board of Technical Education, B.E. degree in Electrical & Electronics Engineering from the University of Goa and the M.E. degree in Power & Energy System Engineering from the University of Goa.

Since 2013, he has been an Assistant Professor with the General Engineering Department, Goa University, Shree Rayeshwar Institute of Engineering & Information Technology. He has authored more than 8 Research Publications & along with 2 Research Paper Publications in "IEEE Xplore". His research interests include EMP (Electro-Magnetic Pulse) Applications in Modern Warfare, Free Energy Technology, & Advancements in Renewable Energy Sources & Directed Energy Weapon (DEW) Technology.