# DEPARTMENT OF MECHANICAL AND INDUSTRIAL ENGINEERING

**Controlling of a Mobile Robot by Vision system**

(MECH 6631)

Submitted to:

## Dr. Gordon Brown

May 06, 2006


Prepared By:

| | |
|---|---|
| **Mohamad Wassim Sharbek** | |
| **Ahmad Ibrahim** | |
| **Md. Ruhul Amin** | |

## ABSTRACT

The recent trend of changing technology, production process and its control are pushing the world in a multi dimension of industrial revolution. Cost effective and energy efficient design, integrated manufacturing system, competitive market and continual changing of end user behavior are the prime factor for industrial production.

Invention of new technology, implementation of more efficient production process, applying lean manufacturing, six sigma and 5's can play an important role for industrial sustainability. Also automated production process, utilization of robotics and applying image processing & vision control can bring down production time and cost exponentially. Furthermore, Precision work, laser control and mass production are almost impossible without robotics or automatic control system.

Controlling a robot using image processing or vision technique is now been proven the most efficient control system. Euclidean control or magnetic control can't detect the obstacle clearly

# ACKNOWLEDGEMENT

We would like to mention our deep appreciation to Dr. Gordon Brown for giving us the opportunity to work on exciting Vision Control Mobile Robot project.

Mohamad Wassim Sharbek
Ahmad Ibrahim
Md. Ruhul Amin

# TABLE OF CONTENTS


1.0 Introduction---------- : 5

2.0 Objectives---------- : 5

3.0 Hardware ---------- : 5

4.0 Methodology ---------- : 5

5-0 Algorithms and procedures ---------- : 6

5-1 The main function ---------- : 6

5-2 Hunt color function ---------- : 8

5-3 Read label function---------- : 10

5-4 Draw line function ---------- : 10

5-5 Find virtual point function ---------- : 10

5-6 Find direction function ---------- : 14

5-7 Move to function ---------- : 15

5-8 Robot control functions: ---------- : 16

5-9 Other functions ---------- : 16

6.0 Future works ---------- : 17

7.0 Conclusion ---------- : 18

8.0 References ---------- : 18

## 1.0 INTRODUCTION

Robot is the integrated part of the modern industries whose performance depends upon controlling method. Computer Vision system helps the robot to work independently. A programmable mobile robot, control by vision technology can perform different types of sophisticated work on an independent environment. An independent vision control motor like mobile robot is designed as a part of the project for MECH - 6631 course. RTX and SDK software are used to control the motor of the robot and vision system respectively. C language is used for concatenation, interfacing and controlling the complete project. The report covers:

- Objective
- Methodology.
- Control procedure.

## 2.0 Objective:

Designing, constructing and controlling of a motor like mobile robot which will picks an object from a specified location and putting it in another position, avoiding all the obstacles on its way by using vision technology.

## 3.0 Hardware:

Three 5 volt DC servo motor are used for constructing the system. Two motor controls the motion (forward, backward, left and right) and one motor for grapping the object. A 640X480 web-camera is used to detect the object and obstacles.

## 4.0 Methodology:

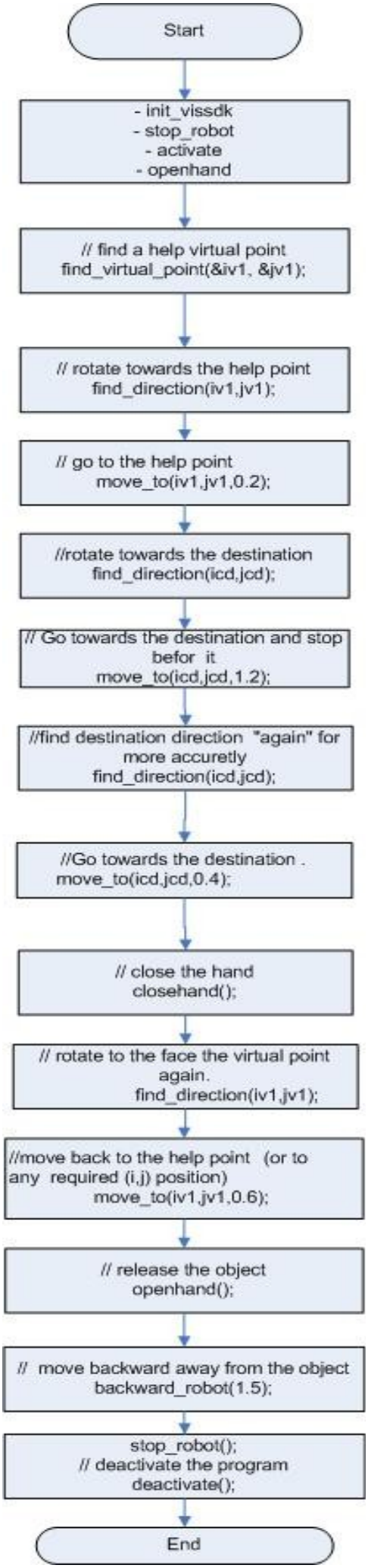
The car like mobile robot works independently within its work-space. The object and obstacle might be any shape but specified in color. The robot will distinguish the object and obstacle and locate the position by their color by using computer vision system. The robot might be faced to any direction but aligns itself during operation and moves smartly by using PID Controller towards the object avoiding the obstacles by assigning a virtual help point and then holds the object and moves it to any specified position.

## 5-0 Algorithms and procedures:

### 5-1 The main function:

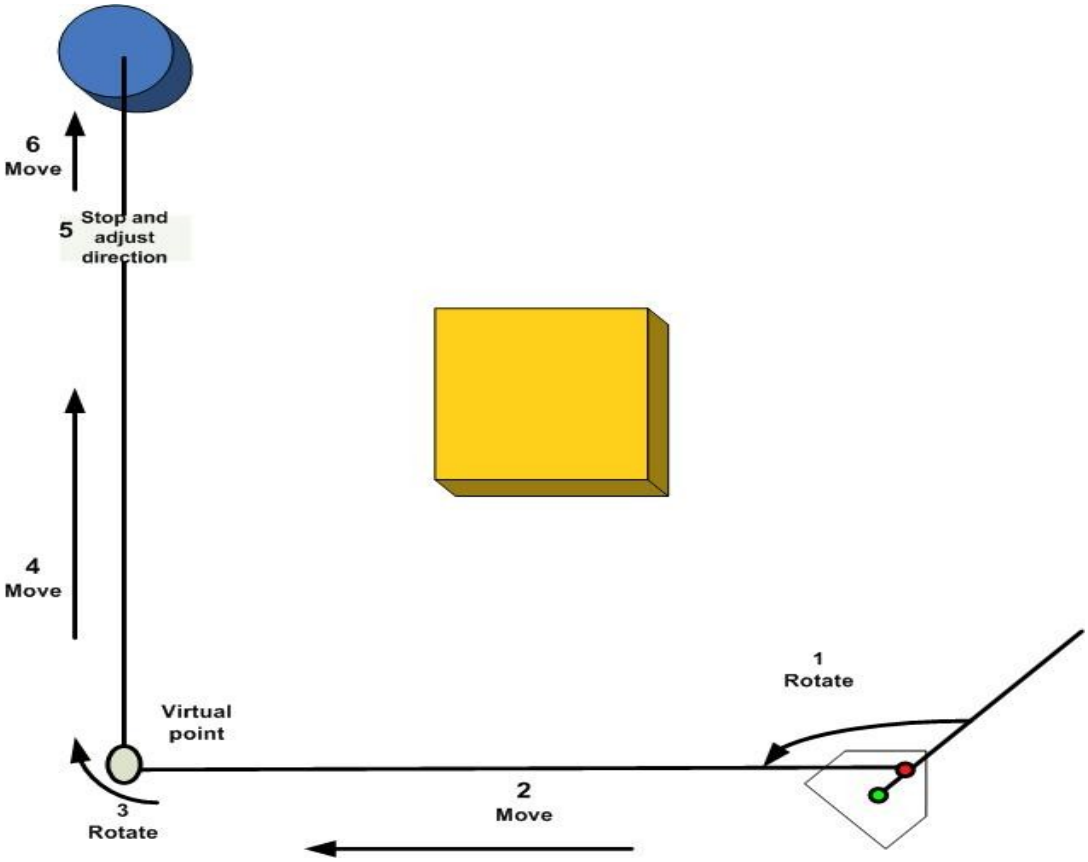
The following flowchart describes the main function of the program. Explanation about each function comes latter.

It's essential to take a photo to the robot, obstacle and the destination object before start working to analyze the colors and find out the RGB range for each object and then provide the RGB range value in the program to be used by the "Hunt colors" function (see page 8 )




- Initialize vision SDK and setting the motor pulse width on zero. Also setting the type and size of the images and allocate the memory to the images, and open the robot hand.
- To avoid the obstacle, a help virtual point (iv,jv) will be found by using a robust algorithm (See page 10).
- After the help point is determined, the robot will rotate to face this point by using the "Find direction" function (See page 14..)
- By "Move to" function, the robot now will move forward towards the virtual point and stop at it. (See page 15..)
- The robot now is ready to go to the destination without hitting the obstacle, but firstly it will rotate itself to face the destination.
- The robot moves forward towards the destination ,and for more accurately the robot stops before getting the distention in a short distance to correct its direction if it's not totally correct.
- If the robot is not facing the destination accurately and correctly, it turns right or left to adjust itself.
- The robot moves again towards the destination and stops when the destination object is between the robot hands.
- The robot hand is closed to catch the destination object.
- After catching the object, the robot rotates to face the assigned virtual point or to any required position.
- The robot goes towards the virtual point and stops at it.
- The hand of the robot is opened to release the obstacle.
- The mission now is successfully completed and the robot moves back away for the obstacle and stop far away from it ready for another mission.
- The motor pulse widths are set to zero to totally stop the motors and the program is deactivated and the memories is freed.

The next figure shows how the robot gets the destination from the initial position.




Before illustrating the other functions of the main program, we will illustrate some of the essential functions that are considered as backbone of the other functions.

### 5-2 Hunt color function:

***int hunt_colour(image*rgb, int Bl, int Bh, int Gl,int Gh, int Rl,int Rh, int* ih, int* jh)***

**The objective** of this function is to store ,hunt, a position (ih, jh) of any pixel ,usually the last on, inside the required colored object. This pixel is used by another function ,read_label function, to read the label of this object when it is labeled.

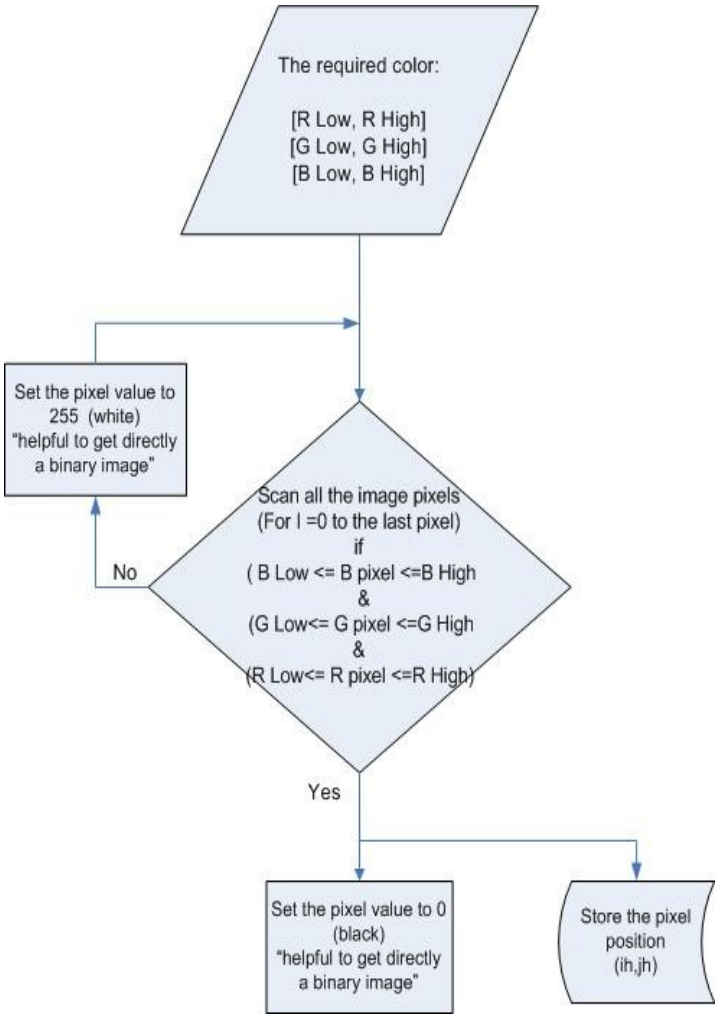
To hunt the 4 colors, we need to scan the image 4 times, so we developed this function to hunt 4 colors in one image scan. This function works in the

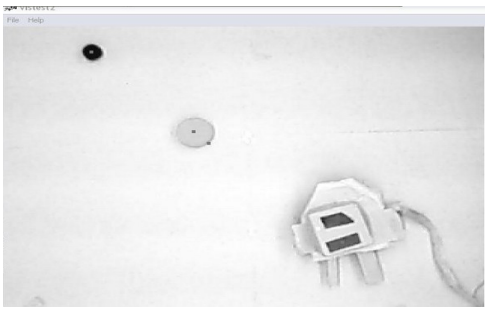
same way that hunt 1 color works, but it saves operation time. The developed function has the following form :

***int*** ***hunt_colours(image*rgb,***

***int Bla,int Bha,int Gla,int Gha,int Rla,int Rha,int* ia,int* ja,***
***int Bld,int Bhd,int Gld,int Ghd,int Rld,int Rhd,int* id,int* jd,***
***int Blh,int Bhh,int Glh,int Ghh,int Rlh,int Rhh,int* ih,int* jh,***
***int Blt,int Bht,int Glt,int Ght,int Rlt,int Rht,int* it,int* jt )***

The next flow chart illustrates how "hunt colors" function works. While the other figures show images before and after hunt color is applied.





**5-3 read label function:**

***int read_label(i2byte *nlabel, image *label ,int I ,int j)***

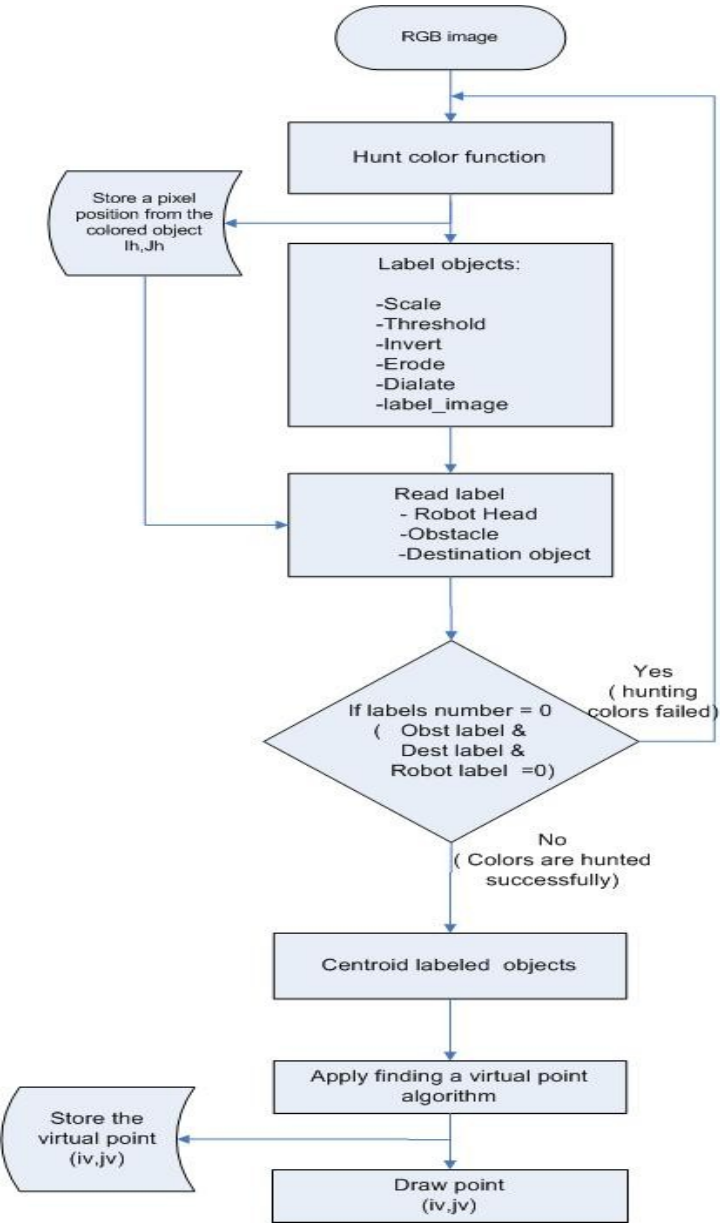
This is a sample function read the label value at co-ordinate (i,j).

**5-4 Draw line function;**

***int draw_line (image *a, int ip1, int jp1, int ip2, int jp2,int value,***
***double *alfa, double* distance)***

By this function not only we can draw a line between the points (*ip1, jp1)* *( ip2, jp2)* but also we get the length of the line and the angle between the line and the axe J. We depend on this function to rotate our robot and to move it for certain distance.

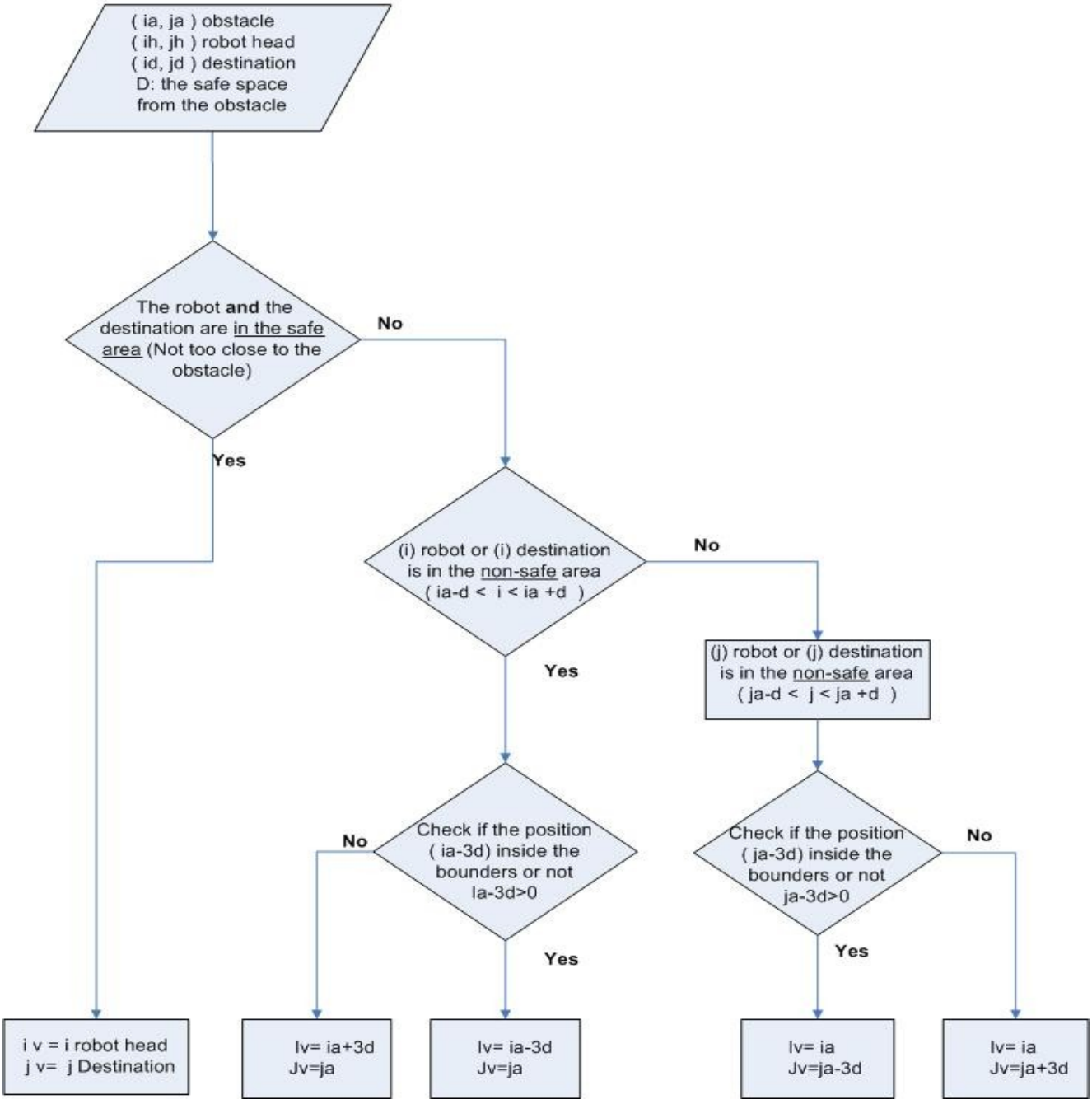
**5-5 Find virtual point function:** ***int find_virtual_point(double *iv,double *jv)***




This flow chart illustrates the "find virtual point" function that the robot will go to avoid hitting the obstacle.

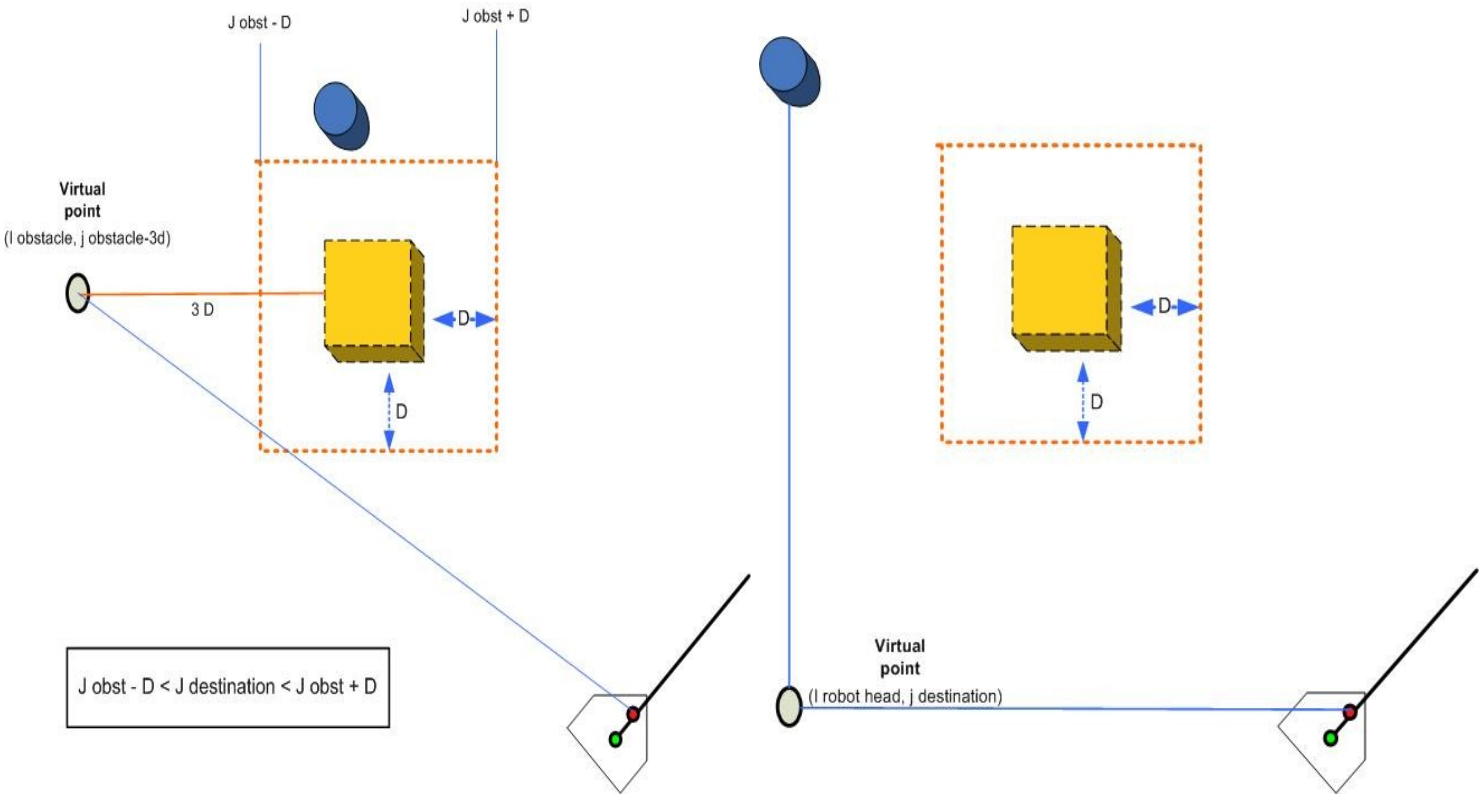
- In brief description, After the colors are hunted, a position of a pixel from the colored object is stored.
- The image is labeled and each object gets a label number.
- The stored pixel position that indicates to particular color is passed to read label function to recognize the colored object.
- If label numbers for the obstacle, destination and the destination are got, the process successes, otherwise the color hunting is not accurate so the previous actions are repeated.
- By passing the label number of each object to the centroid function, the centers of the obstacle, destination and the robot head are obtained. At this point, the obstacle and destination centers are stored to be used during all the operation because they are not changed.
- The centers are used by the algorithm of finding the virtual point which is explained next

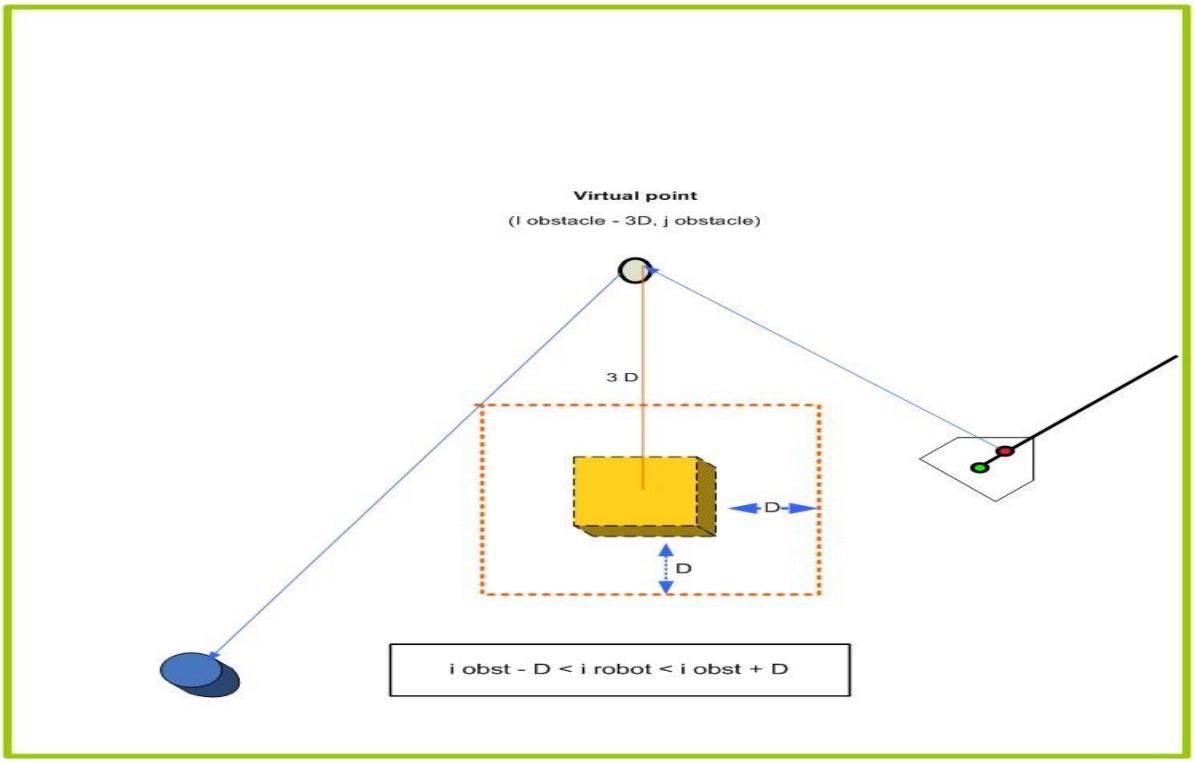
The next flowchart explains the algorithm of finding the virtual point.

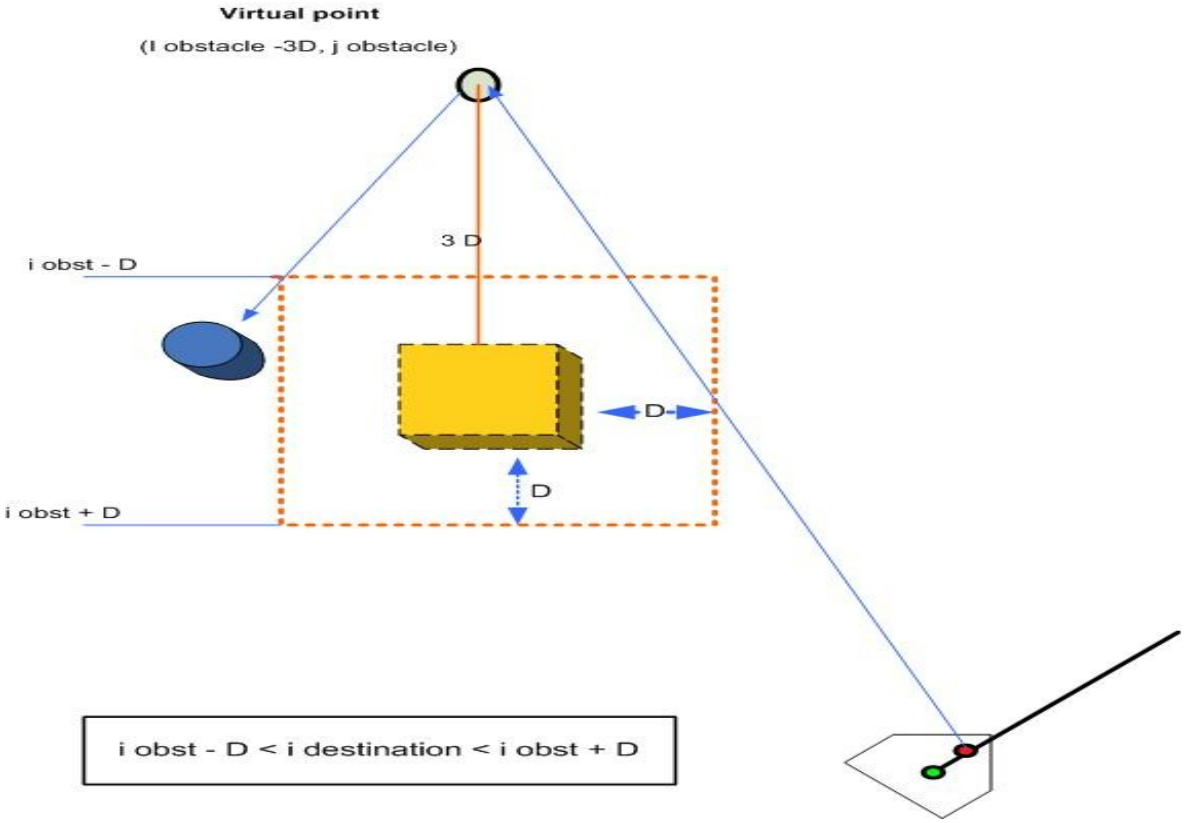


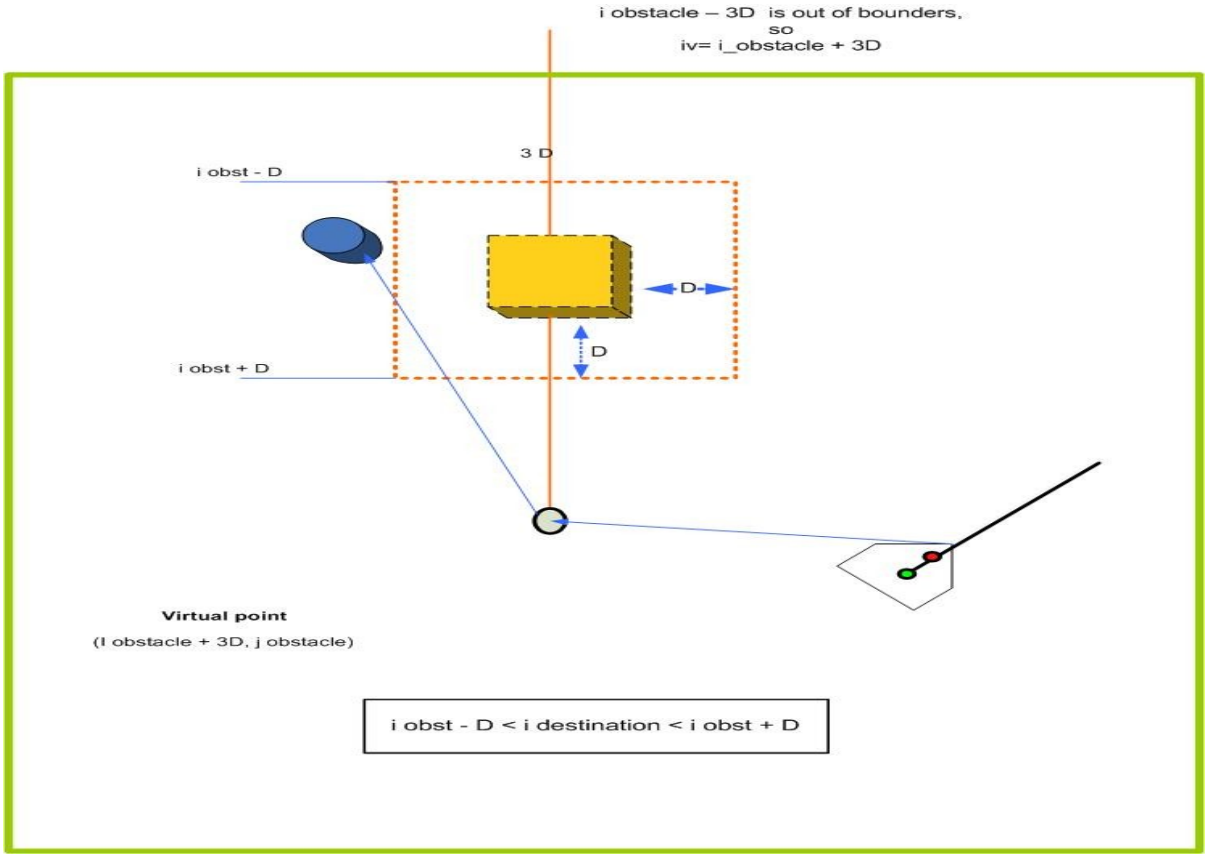

In this algorithm the robot arena is divided into safe area and non-safe area. The robot or the destination is in the non-safe area if they are located in the area [i obstacle – D , i obstacle + D ] or [ j obstacle – D , j obstacle + D ] , where D is required distance to keep the robot far away from the obstacle. This algorithm is robust and can be easily modified if a new not desired situation shows up.

The next figures shows different scenarios to find the virtual point according to this algorithm.



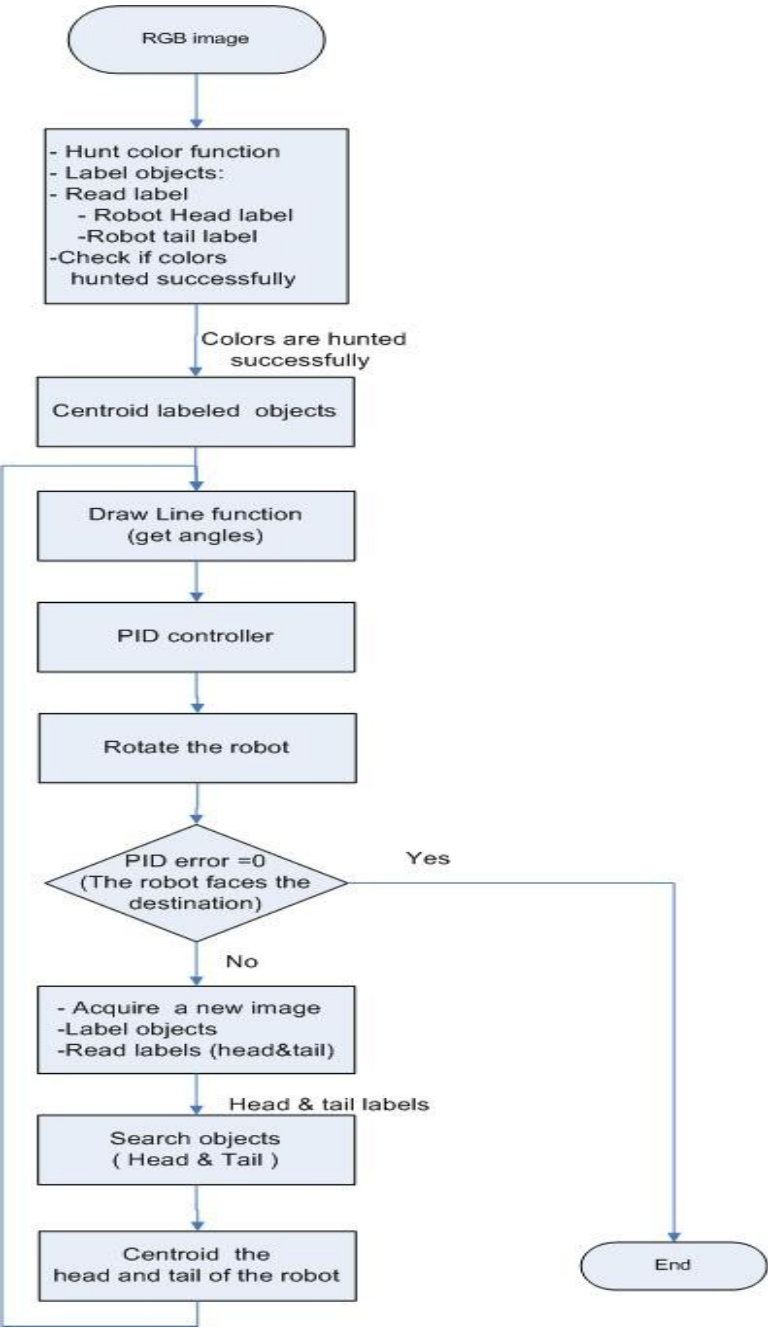




Virtual point
(I obstacle -3D, j obstacle)
3 D
i obst - D
D
D
i obst + D
i obst - D < i destination < i obst + D


i obstacle – 3D is out of bounders,
so
iv= i_obstacle + 3D
3 D
i obst - D
D
D
i obst + D
Virtual point
(I obstacle + 3D, j obstacle)
i obst - D < i destination < i obst + D

**5-6 Find direction function:** ***find_direction(double i_dir,double j_dir)***

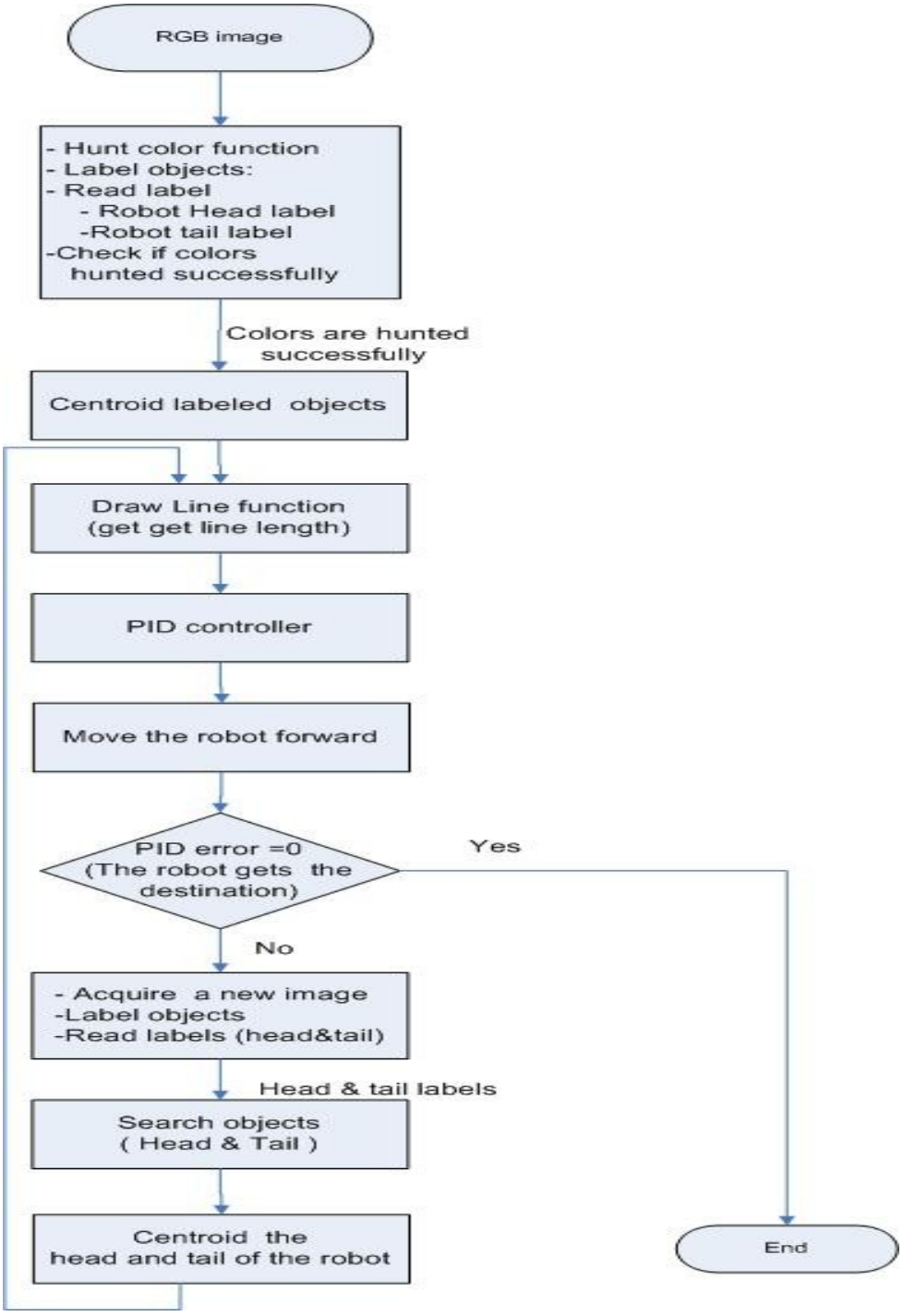
By this function, the robot turns right or left to face its destination (the destination or the virtual point). The main idea here is that a line is drawn between the two points of the robot (the head and the tail) and another line is drawn between the robot head and the destination. Each line makes an angle with the axe J. By passing these two angles to a PID controller and taking the difference as a PID error, the PID will control the motors until the difference between the two angles is zero and that means the robot faces the destination directly. The following photo shows the two lines of this function, and the next flow chart explaining how this function works:




**5-7 move_to function :**

***move_to(double i_dir,double j_dir,double req_distance)***

This function is used when the robot faces the required destination. The idea behind this function is to draw a line from the robot head into a destination **(i dir, j dir)** and the draw line function will provides the length of this line. By passing the line length to a PID controller and compare it by the *req distance*, the PID will control the movement and stop the motors when the root gets the required position. The next flow chart shoe the working of this function.




**5-8 Robot control functions:**

These functions work directly with the motor control function that controls the motors.

Stop robot, rotate robot, neutral robot forward robot, backward robot, open hand and close hand function are the robot control robot. All of these functions are simples and easy to be built. The main idea behinds these

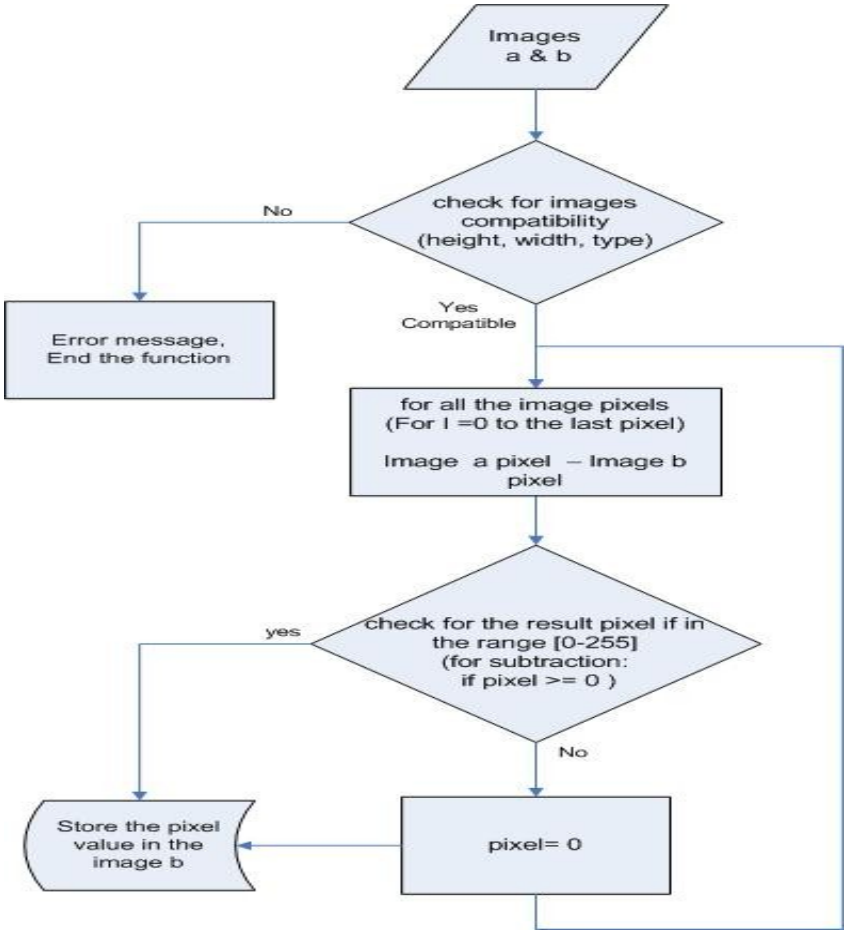
functions is to set the pulse width values to suitable values ,and then send the value to the motor control function.

### 5-9 Other functions:

There are some other functions that we built but we did not use them in this project. As an examples of these functions:

| | |
|---|---|
| *int subtraction(image*a, image*b)* | image b =image a-image b |
| *int addition(image*a, image*b)* | image b =image a+ image b |
| *int AND (image*a, image*b)* | image b =image a & image b |
| *int OR(image*a, image*b*) | image b =image a or image b |

The following flowchart illustrates how the subtraction function works.




## 6.0 Future work:

Working on mobile robots and vision systems open doors to develop many interesting applications, so this project can be developed in many ways. Due to shortage of time and our limitation, we could not applied spline to move the

robot more smoothly with shortage possible time. Selection of virtual point is good for avoiding the obstacle but selection of proper spline might be more time as well as cost efficient. Therefore, in future, if we get opportunity, will try to control the robot by applying spline and compare it with our present project. Analyzing the color rage and store it automatically, will reduce the working time drastically, giving completely automated process. Therefore, effort can be given to develop such program to support it.

Furthermore, using many colored obstacles and multi destination will make the project more challenging and interesting, and the robot can go around these obstacles to get a certain destination. Also, object -shape recognizing can be used with colors to distinguish different objects.

Controlling the robot remotely provides more flexibility to this project. This can be achieve either by going to the RF wireless world, or by the internet protocols (TCP, UDP,IP, …) to control and to observe the robot from far distances through the internet. Moreover integrating the internet protocol with the wireless 802.11 help doing more advanced missions.

## 7.0 Conclusion :

A car like mobile robot is built and controlled by Vision technology with the help of RTX and SDK software. A complete program using C language is developed for image processing, analyzing and controlling the robot. Most of the project work focus on, how to avoid the obstacle. For convenient

movement and avoiding the obstacle, the robot will search, align and move to a *virtual point*. This virtual point completely depends on the image of the colored object, obstacle and robot. Accuracy of the control depends upon the proper selection of virtual point which is completely depends upon the detection of color. But in the work place due to the effect of reflection and noise, it is very hard to detect real color. Therefore, a range of RGB value are analyzed and put manually. For simplification, to avoid derivative part of PID controller, PI controller are used for proper alignment of the robot with the virtual point as well as with the object.

## 8.0 Reference:


1. Class note, Mech -6631 , " Dr. Gordon Brown".
2. Modern Control System, " Richard C. Dorf".
3. Control system design, " Chi-Tsong Chen"
4. http://www.doc.ic.ac.uk/~ih/doc/par/
5. http://www.lvr.com/jansfaq.htm
6. http://www.lvr.com/parport.htm