# Barcode Scanning Bot for Library

GROUP 404

| | |
|---|---|
| 1: Sravan Patchala(group leader) | 14D070012 |
| 2: Saurabh Gangurde | 14D070009 |
| 3: Yogesh Mahajan | 14D070022 |
| 4: Nischal Agrawal | 14D070049 |

# INDEX

# Abstract


## **Main Idea** :

To scan barcodes on books to help in maintenance of library. Robot with camera mounted on it follows the white line drawn parallel to shelf to scan barcodes.Program sorts barcodes in absent or extra category to help to find out misplaced books. Program can also generate list of barcodes for new shelf.

## **Hardware Used** :

- FireBird V Robotics development kit
- Webcam
- Xbee wireless module

This report contains detailed explanation of project and structural organisation of project.

# Introduction

A book placed on the wrong shelf is as good as a lost book.
In library large number of books are removed from their placed but many of them are not placed back to their proper shelf.According to library system book is at it's place but in reality it is not.Library staff can't check for every book every day,but a robot can do this.

Robot can apply different methods to get wrongly placed book,but we can achive this without modifing too much in existing system which runs on barcodes.Barcode scanning Bot follows a white line drown on shelf in front of books.Camera mounted on bot passes video to master computer for detecting barcodes and decoding them.This scanned barcodes are compared with database of shelf to find out misplaced books.Program can also genetate database for new shelfs.Program generates seperate text files to store the result of each scan.This files can be used to access misplaced books.

This type of arrangement can be very helpful in libraries having large number of books.This system does not need too much modification in current software system of library.Only some changes are required in structure of shelfs.

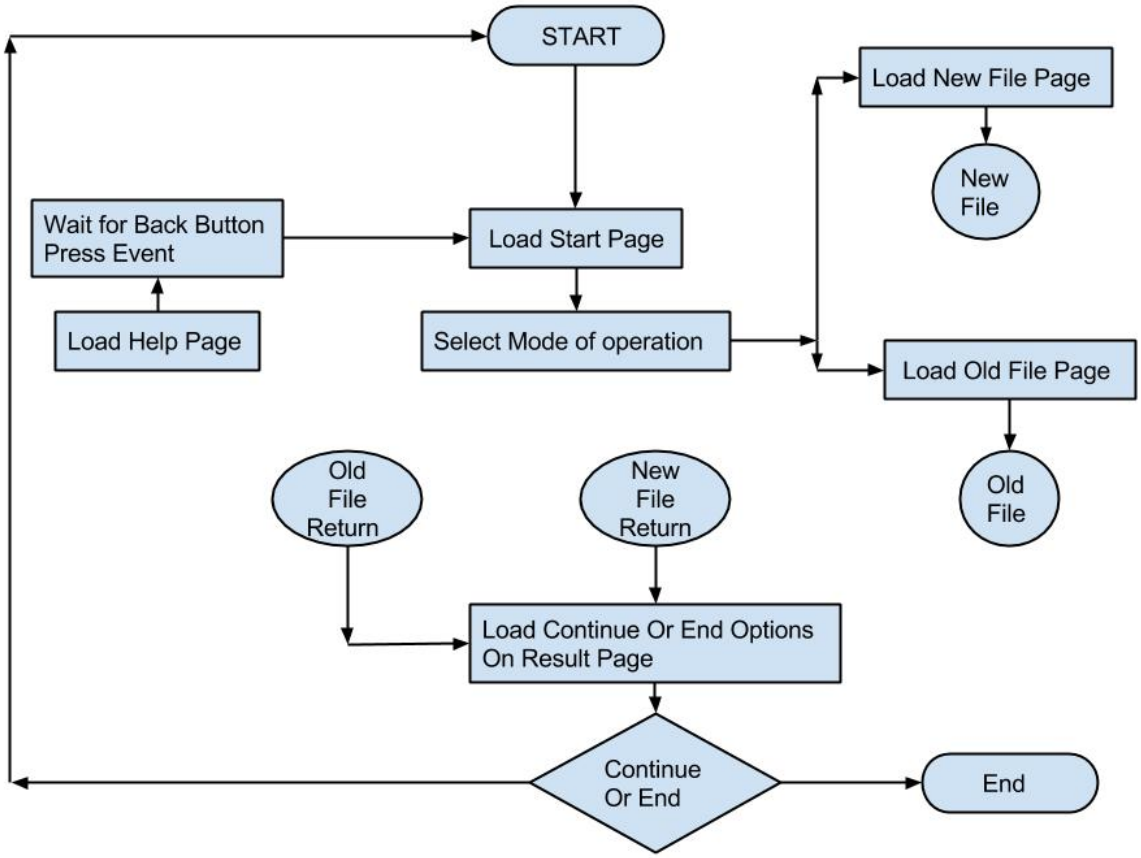
At the present stage program uses EAN 13 barcodes which are very common and present on almost everything we buy.Program can be modified for other codes very easily depending on requirements.

# Technical Details

- C++ compiler compatible with OpenCV. (We used Code Blocks 13.12)
- Library used for image processing- OpenCV
- Interface between computer and bot : XBee wireless serial transmission module.
- Robot used - Firebird V robotics development kit(Atmega2560).
- Header file used for communication through serial port - Windows.h
- Camera used 1.5 Mp USB webcam(can be replaced with wireless/ or IP camera).
- Atmel studio(6.0).

# Design of Program

## Main Program Algorithm




There are two modes of operation as explained above,New File and Old File.Main menu contains 4 options

1: New File Mode

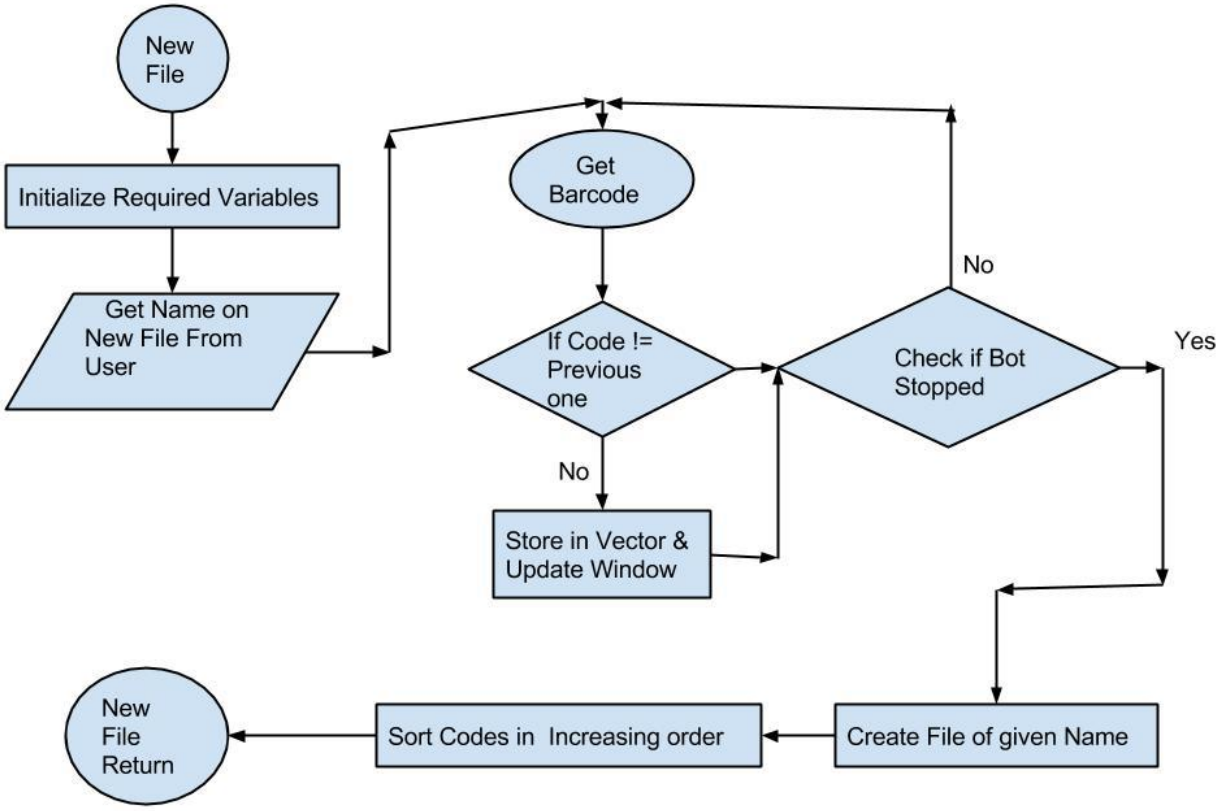
2: Old File Mode

3: Help

4: Quit

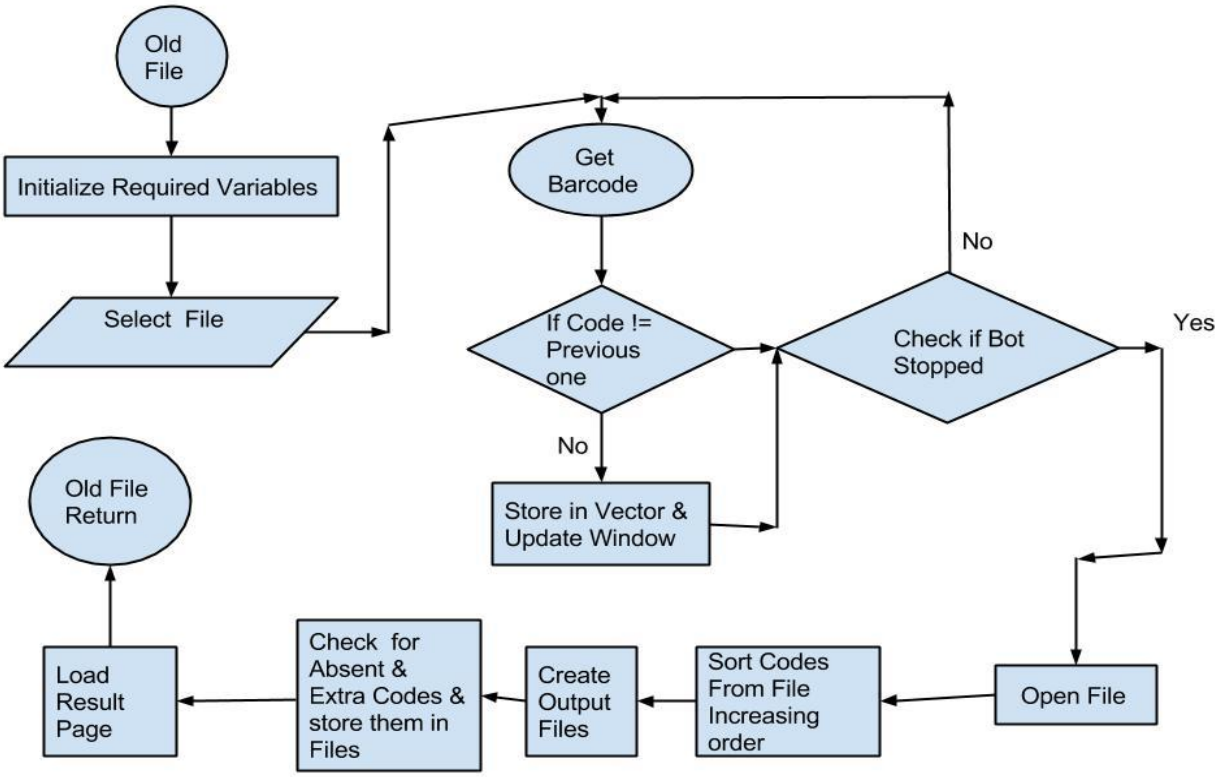
Then control is transferred to respective module according to mode selected.

# New File Mode Algorithm




New file mode is useful in creating new database, e.g adding new shelf to existing database. User have to enter name for file to be created. After initializing required variables code starts a loop to read barcodes. Loop calls a function which causes bot to start following line slowly and returns a barcode which is saved in vector. This part also creates file to store result and then returns to mainstream with continue or end options.

# Old File Mode




Old file mode is useful to check extra or absent items from given database.

This part first shows list of available database list which is stored in "fileList.txt". Then starts getting barcodes same as in new file mode.At the end of loop it loads data from file selected from user and sort scanned data in three categories:

1) Absent Barcodes : stored in "selected_file_nameAbsent.txt"
2) Extra Barcodes : stored in "selected_file_nameExtra.txt"
3) Present Barcodes : Not stored separately as they are correctly placed.

After sorting all files are saved and result is shown and redirected to end or continue option.

# Implementation of Program

Whole program is divided in 4 files.

**1: main.cpp**

Contains main flow for whole program including user interface and files opening & saving operations.This also contains functions for serial communication with bot,which starts and stops barcode detection and decoding.

**2: file_io.cpp**

This file contains functions for creating names of output files and loading database in program.

List of Functions

a) long long getNumFromString(string str)

generates numerical long long value of provided string.

b) vector<long long> getSortedBarcodesFromFile(string fileName)

Loads barcodes from stored database in increasing order in vector.Sorting is done so that user can input database manually,which may not be sorted.

c) string createOutputFileExctraName(string inputFileName)

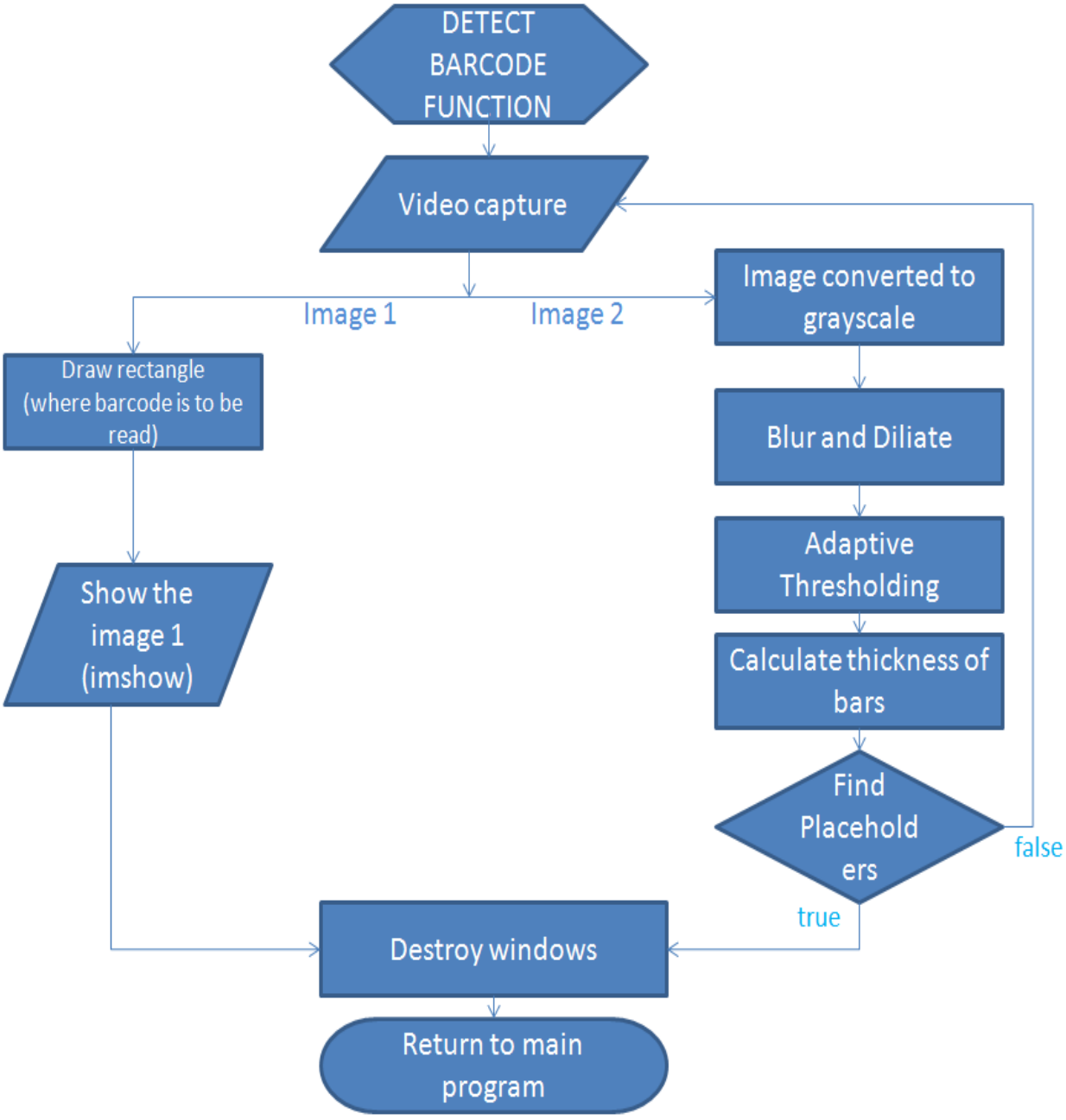
Creates name of file for storing result of extra barcodes.

d) string createOutputFileAbsentName(string inputFileName)

Creates name of file for storing result of absent barcodes.

## 3: detect.cpp(detects barcode by placeholder)

**FLOWCHART OF DETECT FUNCTION**

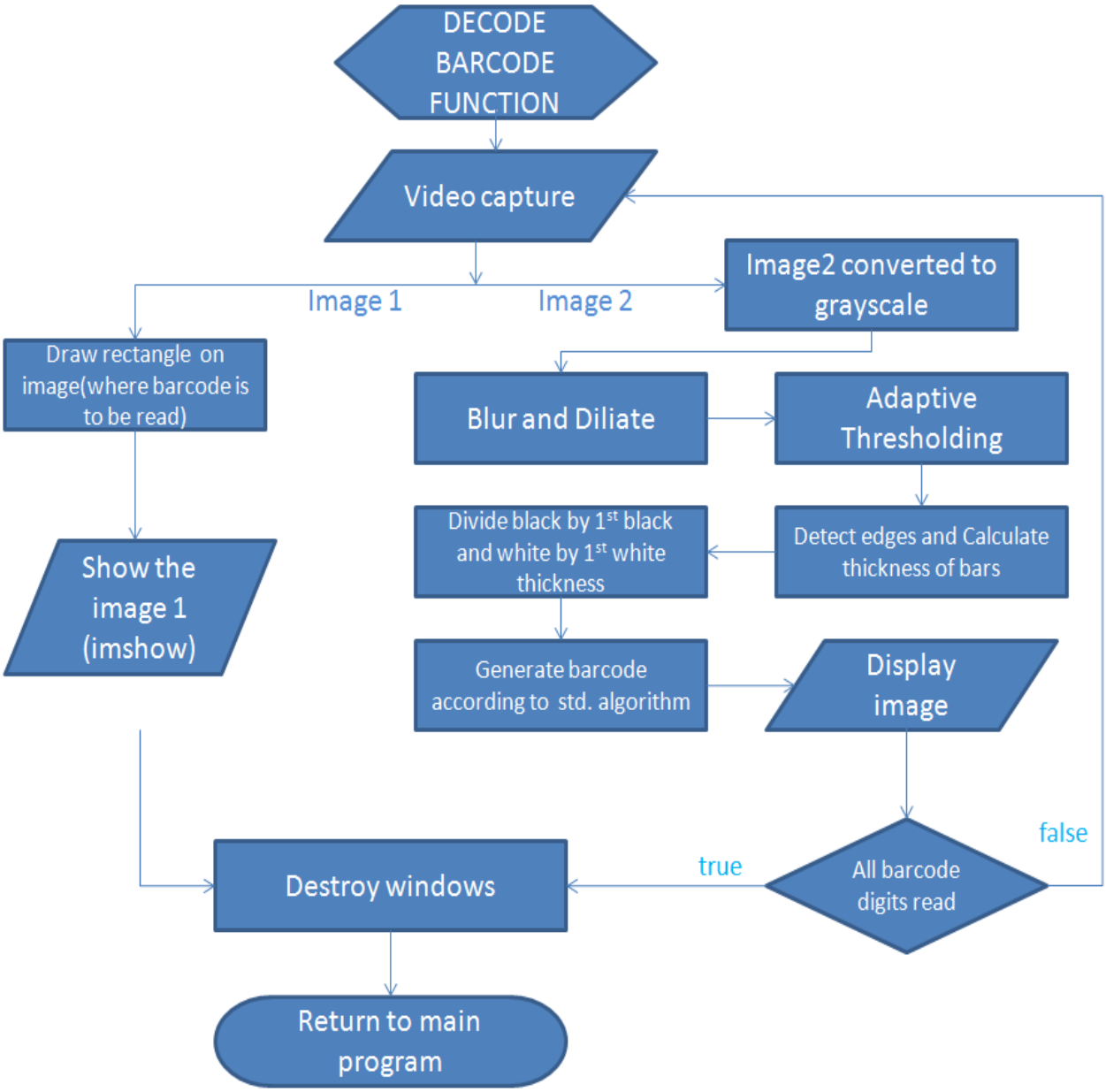


Function used is_barcode():

barcode is detected using staring placeholders as shown in picture::

## 4: decode_barcode.cpp

**FLOWCHART DECODE BARCODE**



## How barcode works?How encoding and decoding is done?

To encode an EAN-13 barcode, the digits are first split into 3 groups; the first digit, the first group of 6 and the last group of 6. The first group of six is encoded using a scheme whereby each digit has two possible encodings, one of which has even parity and one of which has odd parity. The first digit is encoded by selecting a pattern of choices between these two encodings for the next six digits, according to the table below. (Unlike the other digits, the first digit is not represented directly by a pattern of bars.). The white bar corresponds to one(1) and the black bar corresponds to zero(0).

| First digit | First group of 6 digits | Last group of 6 digits |
|---|---|---|
| 0 | LLLLLL | RRRRRR |
| 1 | LLGLGG | RRRRRR |
| 2 | LLGGLG | RRRRRR |
| 3 | LLGGGL | RRRRRR |
| 4 | LGLLGG | RRRRRR |
| 5 | LGGLLG | RRRRRR |
| 6 | LGGGLL | RRRRRR |
| 7 | LGLGLG | RRRRRR |
| 8 | LGLGGL | RRRRRR |
| 9 | LGGLGL | RRRRRR |

| Digit | L-code | G-code | R-code |
|---|---|---|---|
| 0 | 0001101 | 0100111 | 1110010 |
| 1 | 0011001 | 0110011 | 1100110 |
| 2 | 0010011 | 0011011 | 1101100 |
| 3 | 0111101 | 0100001 | 1000010 |
| 4 | 0100011 | 0011101 | 1011100 |
| 5 | 0110001 | 0111001 | 1001110 |
| 6 | 0101111 | 0000101 | 1010000 |
| 7 | 0111011 | 0010001 | 1000100 |
| 8 | 0110111 | 0001001 | 1001000 |
| 9 | 0001011 | 0010111 | 1110100 |

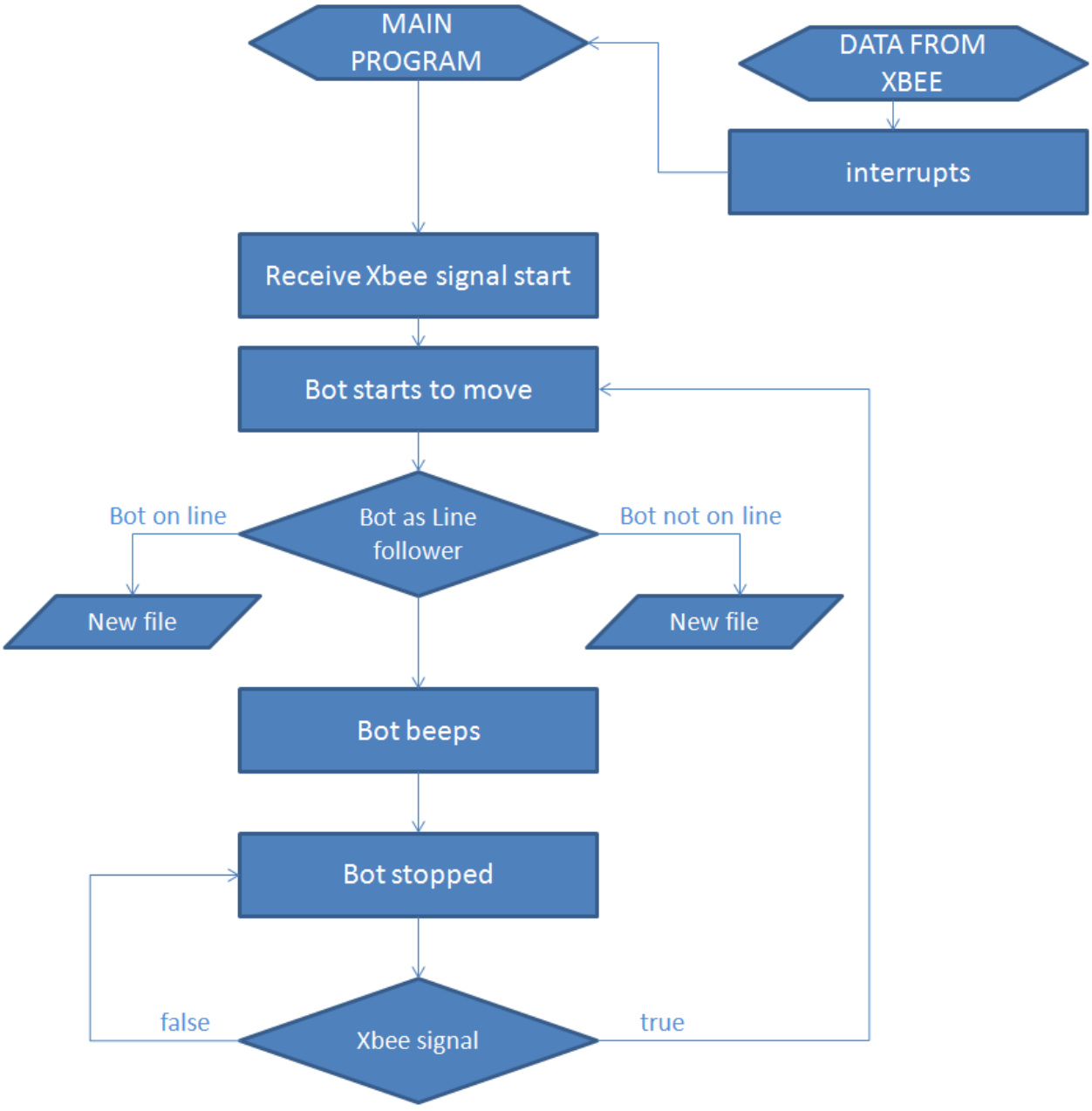
**Note**: Entries in the R-column are bitwise complements (logical operator: negation) of the respective entries in the L-column. Entries in the G-column are the entries in the R-column in reverse bit order. See pictures of all codes against a colored background.

**FUNCTIONS IN "decodeBarcode.cpp"**

```
int number_generator(int mini_array[])
Converts an array of length 4 called mini_array to a number
equivalent.

int l_convention(int mini_array[],int counter)
Used to generate the number for the bit pattern in L convention.

int r_convention(int mini_array[],int counter)
Used to generate the number for the bit pattern in R convention.

int g_convention(int mini_array[],int counter)
Used to generate the number for the bit pattern in G convention.

bool is_edge(Mat &image,int x,int y,int previous_pixel)

void decodeBarcode(short int *barcode,int n)

Captures frames from cap object  of VideoCapture class
        cap.read(image),cap.read(image1);

OpenCV function to change color image to gray scale
        cvtColor(image,image,CV_RGB2GRAY);


OpenCV function to get kernel of given the size
        Mat element = getStructuringElement(MORPH_RECT ,Size(
2*dilation_size + 1, 2*dilation_size+1 ),Point( dilation_size,
dilation_size ) );

Thresholds image according to OpenCV algorithm.Dynamically decides
the threshold value of the image by calcuating average (or weighted
average(gaussian))of the pixels.
        adaptiveThreshold( image,
image,255,CV_ADAPTIVE_THRESH_MEAN_C,CV_THRESH_BINARY,51,5);

Processes the image to reduce noise
        dilate(image,image,element);
```

## 5: bot program

**FLOWCHART BOT PROGRAM**



## FUNCTIONS IN BOT CODE

**void buzzer_on (void)**

Sets pin no corresponding to buzzer high.

**void buzzer_off (void)**

Sets pin no corresponding to buzzer low.

**void lcd_port_config (void)**

Configures Port corresponding to lcd.

**void adc_pin_config (void)**

Configures Pins to Analog to Digital conversion.

**void motion_pin_config (void)**

Sets all pin of motors as ouput and gives initial value low(stop state).

**void port_init()**

Calls all configuration function.

**void timer5_init()**

initialises pins for counting time for interrupts and PWM generation.

**void print_sensor(char row, char column,unsigned char channel)**

Prints sensor value on LCD screen.

**void velocity (unsigned char left_motor, unsigned char right_motor)**

Stores velocity value in OCR5** register.

**void forward (void)**

Sets Pin to 0x06(both pin to drive BOT forward direction).

**void stop (void)**

Sets Pin to 0x06(both pin to drive BOT forward direction).

**void uart0_init(void)**

Sets Pin to 0x06(both pin to drive BOT forward direction).

**SIGNAL(SIG_USART0_RECV)**

Sets interrupt for Xbee input and output.

**void init_devices (void)**

Calls all initializing functions and sets global interrupts.

# Evaluation

This program functions well if all the specified conditions are satisfied properly. Though error tolerance is very low, this project does basics involved in complex method of real time barcode decoding via image processing.Further work of project requires higher mathematics for removing noise in matrix .
Project involves OpenCV techniques to detect and decode barcode. During development we developed code to detect barcode in orientation in real time (also attached as “realTimeBarcodeDetect.cpp”).But we have to reject code as after detecting barcode image resolution is not sufficient to decode it from image.Increasing camera quality may be one solution but it causes large increase in image processing.
Also we have created GUI using “simplecpp”,but we have to remove it due to stability issue of simplecpp with OpenCV in windows. Both works fine in Ubuntu, but we don’t have sufficient supportive framework to use XBee from linux based system (Currently Xbee works only in Windows). For this particular project we are working with UBS webcam, but autofocus IP cam will be better.

# Conclusions and Future Work

This project is very challenging for beginners in image processing. It involves large number of noise handling techniques having large possibilities to develop more and for absolute perfect barcode decoding needs much advanced maths and image processing.
In general complete project can be implemented in real world with some development to make revolutionary changes in library management system.
Not only libraries but same can be implemented in many places such as supermarkets, Godowns , etc.

# References

1) Official openCV documentation: http://docs.opencv.org
2)Barcode detection theory: http://stackoverflow.com/questions/29365813/opencv-2d-barcode-data-matrix-detection
3)eYantra projects: http://www.e-yantra.org/index.php/eyantra/projects
4) EAN 12 barcode standards: http://en.wikipedia.org/wiki/International_Article_Number_%28EAN%29
5) Nex-Robotics(Firebird V referances): http://www.nex-robotics.com/
6) Learning OpenCV by Gray Bradski & Adrian Kaebler book
7) http://opencv-srf.blogspot.in/
8)XBee library: http://www.digi.com/lp/xbee/
9)Project link: https://github.com/sravps7/Library_Book_Barcode_Scanner(git repository).