Barcode Based Automated Parking Management System

Parth Rajeshbhai Zalawadia[1] Jasmin Narbherambhai Surani[2]
[1]M.E. Student [2]P.G. School
[1, 2]VlSI & Embedded System Design Department
[1, 2]Gujarat Technological University, Ahmedabad, Gujarat, India

Abstract— Barcode Based Automated Parking Management System is implemented in this paper. The automatic car parking system using barcode is an interesting project which uses barcode as its heart. This project is designed for car parking. Barcode is a very useful technology in automation of car parking system in a mall/building. It will provide security besides automation of parking though barcode based technology. In this project we are using a particular standard of barcode called UPC-A barcode standard. The project scenario represents two gates are used one at the entry side and other at the exit side which is open/close with the help Stepper motor. Now when a car comes in front of gate, driver has to push a button placed nearby gate. So by pressing this button the barcode chit pops out giving particular no of parking slot to the car. So the entry gate will be open and the car is allowed to park in its designated slot. Now at the exit time driver has to show barcode to the reader machine which is there at the exit gate thus allowing car to leave the parking arena. The advantage of this project is that it saves the high power consumption and no of security persons. The Simulation Results are generated using Matlab 9.

Keywords: Barcode, UPC-A, Graphical User Interface

I. INTRODUCTION

Large organizations today are being challenged to do more with less to deliver higher levels of services at lower costs with fewer resources workforce management systems can help achieve this seemingly impossible goal.

In this system we are allowing a unique barcode for each parking slot. When the car enters in to parking a barcode of a free parking slot is given to that car and then car has to park in that proper slot. At the same time the same space of slot will be booked in our database.

When the car leaves from the parking barcode reader detect the barcode which was given to that car and allows the car to move out. At the same time when barcode scanned the parking slot of that will again be vacant. First of all when car enters in the entrance gate, there is a barcode generator which generates the barcode for vacant space and also tells the number of their parking slot.

For example when the parking slot 1 & 2 are full only then parking slot 3 is allotted to the car. Every time controller automatically checks the slot in ascending order and when it will found that any slot is vacant then barcode generator generates the code for that slot. After generation of barcode sticker the entrance gate will be open and car can move inside parking area. Now when the car wants to leave the parking area, car driver has to detect their barcode sticker to the barcode reader situated at exit gate then and then the exit gate will be open and car will allowed going outside. Once when the barcode scanned at exit gate then controller automatically make that slot vacant and again it will allow for new car. If the circumstances are such that car driver lost his barcode sticker then they can’t move outside the parking area, in such cases the car driver has to contact the security person. The security person has to enter that parking slot number in computer then the exit gate will be open and again that particular slot will be vacant. At the entry and exit gate there is one sensor which is used to close the gate when the car will passed through. Due to this sensor more than one car can’t enter in the parking arena at a same time. When the parking is full then the LED/LCD display situated at the entry gate display that the parking slot is full. Parking System Shown below Fig.1

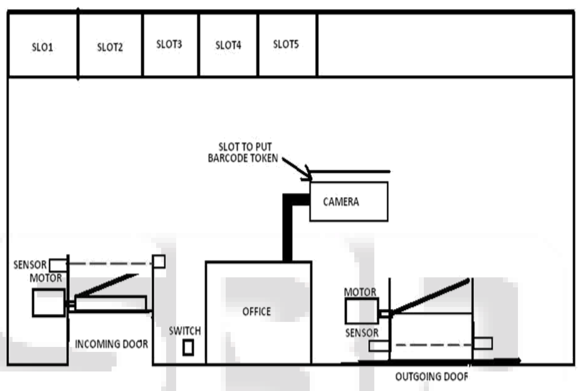


Fig.1 Graphical Representation of Parking System

II. BARCODE STANDARD

A barcode is an optical machine-readable representation of data, which shows data about the object to which it attaches. Originally, barcodes represented data by varying the widths and spacing of parallel lines, and may be referred to as linear or 1 dimensional (1D). Later they evolved into rectangles, dots, hexagons and other geometric patterns in 2 dimensions (2D). Although 2D systems use a variety of symbols, they are generally referred to as barcodes as well. Barcodes originally were scanned by special optical scanners called barcode readers; later, scanners and interpretive software became available on devices including desktop printers and smart phones.

Types of barcode
There are mainly two types of barcodes design available.

– *Linear barcodes*
– *Matrix (2D) barcodes*

A. Linear barcode

The different linear barcode standards are as below.

Symbology	Continuous or discrete	Bar widths	Uses
U.P.C.	Continuous	Many	Worldwide retail, GS1-approved – International Standard ISO/IEC 15420
Code 25 – Non-interleaved 2 of 5	Continuous	Two	Industrial
Code 25 – Interleaved 2 of 5	Continuous	Two	Wholesale, libraries International standard ISO/IEC 16390
Code 39	Discrete	Two	Various – international standard ISO/IEC 16388
Code 128	Continuous	Many	Various – International Standard ISO/IEC 15417

Table. 1: The different linear barcode standards

B. *Matrix (2D) barcodes*

A matrix code, also termed a 2D barcode or simply a 2D code is a two-dimensional way to represent information. It is similar to a linear (1-dimensional) barcode, but can represent more data per unit area.

The different matrix 2D barcodes standards are as below.

Symbology	Notes
3-DI	Developed by Lynn Ltd.
Code 1	Public domain. Code 1 is currently used in the health care industry for medicine labels and the recycling industry to encode container content for sorting.
Color-code	Color Zip developed color barcodes that can be read by camera phones from TV screens; mainly used in Korea.
Dot Code A	Designed for the unique identification of items.
mCode	Developed by Next code Corporation specifically for camera phone scanning applications. Designed to enable advanced cell mobile applications with standard camera phones.
Water Code	High-density 2D Barcode(440 Bytes/cm^2) From Mark Any Inc.

Table. 2: The different matrix 2D barcode standard

C. *Example images*

GTIN-12 number encoded in UPC-A barcode symbol. First and last digits are always placed outside the symbol to indicate Quiet Zones that are necessary for barcode scanners to work properly.

Fig. 2: barcode image

EAN-13 (GTIN-13) number encoded in EAN-13 barcode symbol. First digit is always placed outside the symbol, additionally right quiet zone indicator (>) is used to indicate Quiet Zones that are necessary for barcode scanners to work properly.

Fig. 3: barcode image

D. *Upc-a barcode standards*

UPC barcodes, also known as EAN, UCC-12, are predominantly utilised in North America within the retail industry. There are currently 2 different types: UPC-A, which is a fixed-length 12 digit code; and UPC-E which is a shortened version of UPC-A, consisting of 8 digits. The following discussion will focus upon UPC-A.

UPC-A barcodes employ a 12 digit numerical code. The barcode itself has 2 codes visible: the machine-readable barcode, consisting of a series of bars and spaces; as well as a human readable 12-digit code. The first 6 digits of the code refer to the manufacturer's ID number, increasing the ease at which a product can be aligned to its origin. The next 5 digits are the product's unique number, ensuring that the correct item is read by the software system. The last digit acts as a checksum, allowing the scanner to calculate if the barcode was scanned without mistake. In terms, of calculating the correctness of the machine-readable code by the scanner, a mathematical formula is followed. Each number, ranging from 0 to 9, has its own combination of bars and spaces, making it simpler to decipher the seemingly difficult code. The differing widths of the bars and spaces can be looked at to decipher the numerical code. The thinnest width corresponding to width 1, then there is double that which is width 2, triple width 1 relating to width 3, and width 4, which is four times the width of 1. Or another perspective to view the numerical code is series of binary digits. Each number is assigned a 7 digit binary code. Therefore, in total, UPC barcodes are composed of 95 bits in total. One thing to note is that at the start and end of each barcode, there is a particular bit pattern, 101, indicating to the deciphering program where the barcode is initialized and concluded. There are also guard bars, a particular sequence composing 01010, separating the manufacturer's ID number from the unique product number. It must also be taken to account that the sequences to the left of the guard bars differ to that on the right. The codes corresponding to each number are one's complement when comparing the codes on the left hand side to that on the right.

The pictorial representation of this code is as shown in below figure.

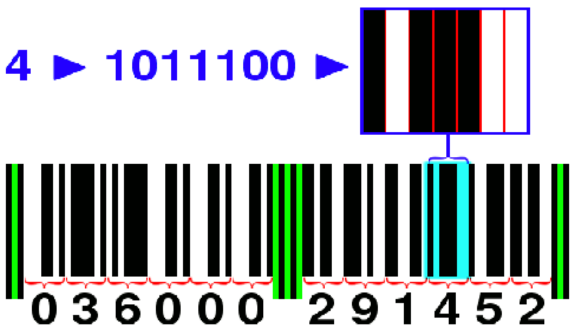


Fig. 4: UPC-a barcode

UPC-A encodes 11 digits of numeric (0 through 9) message data along with a trailing check digit, for a total of 12 digits of bar code data. An example of a typical UPC-A bar code is:

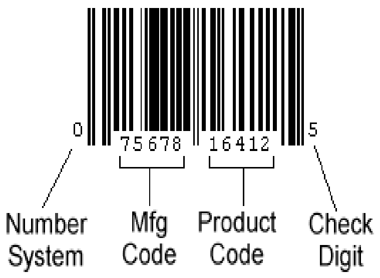


Fig. 5: Detail of UPC-A Barcode

III. BARCODE RECOGNIZATION

The initial GUI is accessed through typing "GUI" in the Matlab command window. From here, the user is able to select the file they wish to decode from a drop down list of files. We have tried to include as many files as possible to show all aspects of our program.

The user is also able to select whether they wish to deblur the image selected and once they are happy with their selection, they are able to initiate the processing of the barcode. Once the barcode is processed, an output window is utilized to show the related numeric code of the barcode. Within this window, the program is also able to tell the user what the error was if it was not able to process the image. The errors are: read error, checksum error, rotation error. The function initially converts the barcode image, into a 1 dimensional array of 1s and 0s. We have designed it such that the 1s correspond to the bars, and the 0s correspond to the spaces of the barcode.

A. *Flow Chart of Reading Image*

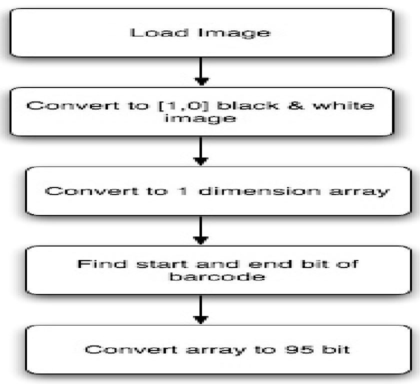

Load Image
Convert to [1,0] black & white image
Convert to 1 dimension array
Find start and end bit of barcode
Convert array to 95 bit

Fig.6: flowchart of reading image

The next important step is to crop the barcode image. The function is designed such that it scans the image, both vertically and horizontally, determining the end values associated with the edge of the barcode. Once these edges are established, the imcrop function within the image processing toolbox is utilized to crop the image to only include the barcode, eliminating any space that surrounds it.

B. *Crop image flow chart*

Since it is known that a UPC-A barcode consists of 95 bits, we convert the found barcode array to the 95 bits, using mathematical formulation. This step is performed to ensure that all the barcodes are uniform and hence, allowing us to determine the widths of each bar and space so we can decode the image into its respective numeric code. Input image with white border

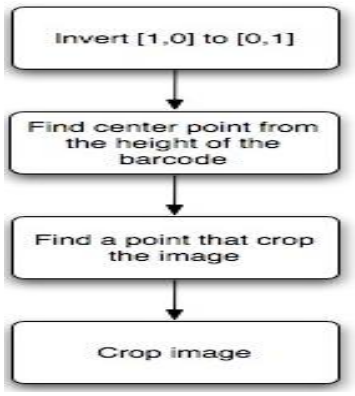

Invert [1,0] to [0,1]
Find center point from the height of the barcode
Find a point that crop the image
Crop image

Fig.7: flowchart of crop image

Fig.8: Input barcode image

Convert image to 1 dimension and as the figure show that the first 37 bit is 1, it indicates that the bar code should start from 38 bit from the figure.

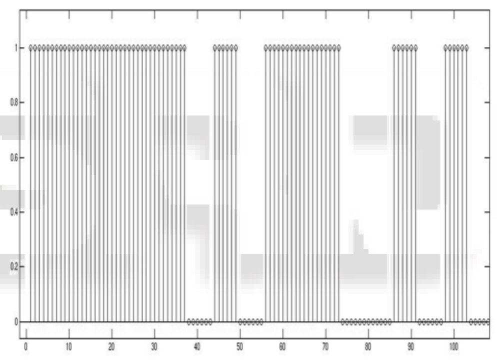


Fig.9: Converted image to 1 dimension

Convert image to 95 bit

After convert to 95 bit, this is the output of readimage.m. Output from this function can go convert.m and get the result of the barcode.

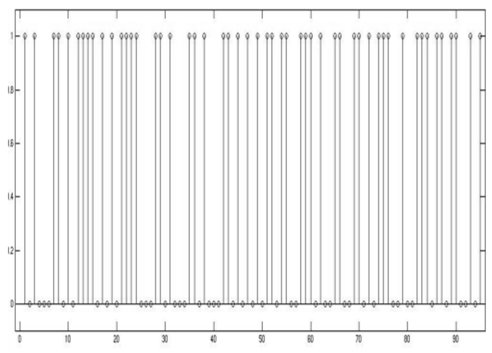


Fig.10: Converted image to 95 bit.

Crop bar code from image this picture indicates that only necessary bar code area crop into new image.

Fig.11: Scanned barcode output

IV. Decoding Process

A. *Original image*

In this section we have taken one original image as shown in fig below.

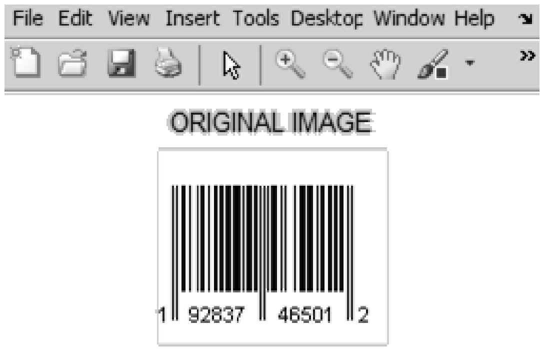


Fig. 12: original image

B. *Inverted image*

Now we have extracted only the barcode region out of the whole image. And after that we have inverted the original image.

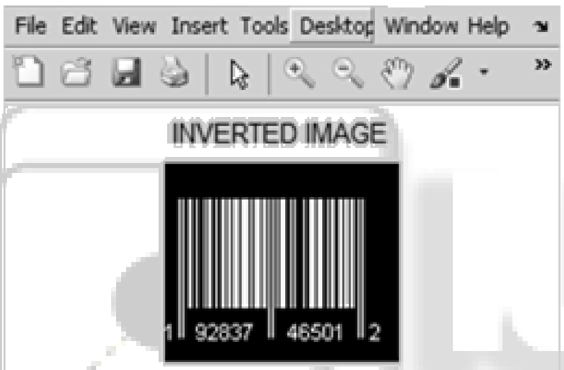


Fig. 13: inverted image

C. *Extraction of Inverted Image*

Now we have cropped the inverted image.

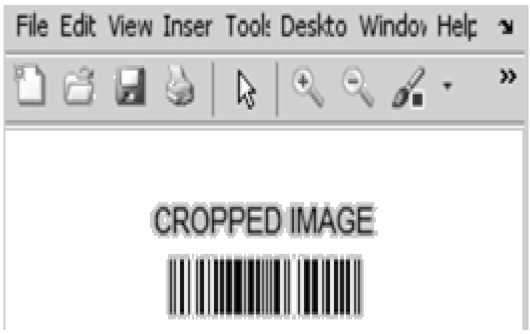


Fig.14: cropping of the inverted image

And then we have extracted the white blocks of that cropped image.

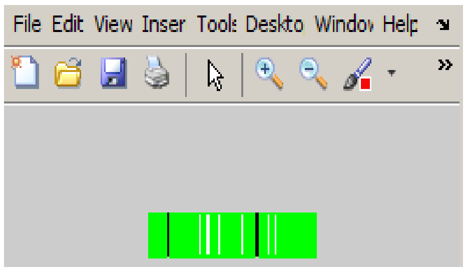


Fig. 15: shows extraction of white blocks

D. *Extraction of Original Image*

After that we further inverted that image to get the original one.

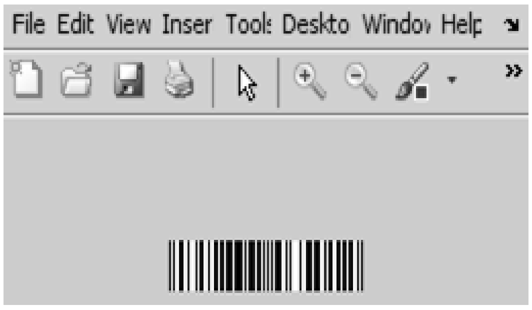


Fig.16: Cropped original image

Now we have again extracted the white blocks out of the original one.

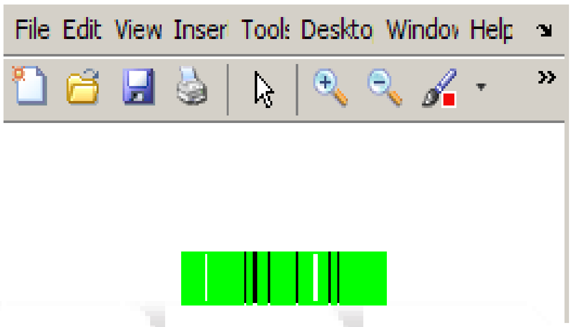


Fig.17: Extracting white blocks out of original

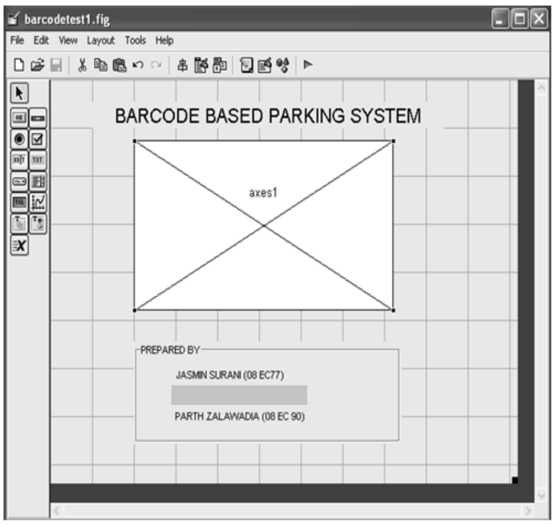
DETECTED BARCODE IS-

1 9 2 8 3 7 4 6 5 0 1

V. Development of Basic Theme

A. *Basic theme of form*

In this form we added a text box to write the title of our project. To show the sample image in the form we use the axes tool for it. To show the other detail we are putting panel, in this we can add other tools also as here we adds a text box in the panel. The basic theme of form is as shown in below figure.



BARCODE BASED PARKING SYSTEM

PREPARED BY

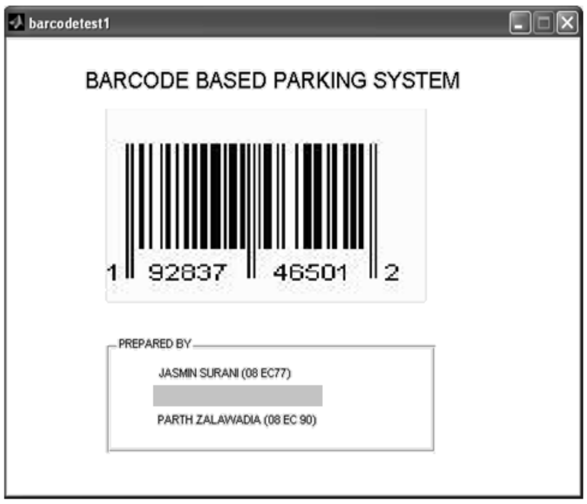
JASMIN SURANI (08 EC77)

PARTH ZALAWADIA (08 EC 90)

Fig.18: Screenshot of complete form

B. *Output of Form*

To run the GUI click the run button we can see the output. The output of the basic theme is as shown in below fig.19



BARCODE BASED PARKING SYSTEM

PREPARED BY

JASMIN SURANI (08 EC77)

PARTH ZALAWADIA (08 EC 90)

Fig.19: Screenshot of output form

VI. Conclusion

In this project firstly we have studied about the MATLAB and its graphic user interface tools like, textboxes, panel, axes, etc. that are related to our project. Then we have learned to make a form in GUI. After learning form we make our basic project theme using GUI tools, and learned various barcode standards out of which we selected UPC-A standard for our project. Then in the last phase we have done coding of barcode on sample images and completed the interfacing part of the model. The results are carried out using Matlab9.

Acknowledgement

Any accomplishment requires the effort of many people and this work is not different. And it is my prime duty to acknowledge the person who directly or indirectly helped me during completion of this project. So I take opportunity to heartily thank our project in charge respected Prof. N. N. Shah for his valuable guidance and touch of inspiration and motivation throughout the project work without whose help the work would not have been in the shape what it is. I also heartily thank my H.O.D Prof. D. U. Shah who greatly helped me in my project work without whose help the work would not have been in the shape what it is. I am also thank full to my co-colleagues for building good team work to make the success of project.

References

[1] E. Ohbuchi, H. Hanaizumi, and L. A. Hock, "Barcode Readers using the Camera Device in Mobile Phones", Proceedings of the 2004 International Conference on Cyber worlds, pp. 260-265, November 2004.

[2] H. Kato and K.T. Tan, "2D barcodes for Mobile Phones," Proceedings of 2nd. International Conference on Mobile Technology, Applications and. Systems, pp. 8, Nov. 2005.

[3] T. Pavlidis, "A New Paper/Computer Interface: Two-Dimensional Symbologies", IEEE Computer, pp. 145-151, Volume 2, June 2000.

[4] T. Pavlidis, et al., "Fundamentals of Bar Code Information Theory", IEEE Computer, pp. 74-86, vol.23, April 1990.

[5] H. Hahn, and J. K. Joung, "Implementation Algorithm to Decode Two-Dimensional Barcode PDF-417", IEEE Computer, pp. 1791-1794, Volume 2, June 2002. Proceedings of the 6th IEEE International Conference on Signal Processing, 2002

[6] T. Pavlidis, et al., "Fundamentals of Bar Code Information Theory", IEEE Computer, pp. 74-86, vol.23, April 1990.

[7] K. Wang, Y. Zou and H. Wang, “Bar code reading from images captured by camera Phones,” 2nd IEE International Conference on Mobility Technology, Applications and Systems (2005).

[8] S. J. Shell hammer, “Novel signal-processing techniques in bar code scanning,” IEEE

[9] Robotics & Automation Magazine, pp. 57–65 (1999).

[10] R. C. Palmer, the Bar Code book: Reading, Printing, and Specification of Bar Code Symbols (3rd ed.), Helmers Publishing, 1995.