Locating, Tracking, and Interpreting Ean-13 Bar Code Waveforms in a Two-Dimensional Video Stream

Jeffrey Adair
Hiram College
AdairJB@hiram.edu

Abstract

A fast method of locating and interpreting bar codes in a two-dimensional video stream is discussed. By using a variation of the hough transformation noise is removed and distortions are reduced. This results in a more robust bar code scanning system.

Problem and Motivation

Bar codes have become ubiquitous throughout all forms of business. It is important to have machines which can read these codes reliably and efficiency. Almost every industry has been fundamentally changed by their invention. The method with which they are currently read, however, has changed little in the last several years. A more efficient way of locating and interpreting these codes is important to their continued use. Using computer vision techniques can reduce the cost and improve the accuracy of the reading systems.

Using computer vision techniques, we will implement a bar code interpreting system which will work in near-real-time. This system will work on a video stream, and will track bar codes which have already been successfully read, to prevent them from being read several times.

Background and Related Work

While there are dozens of forms of bar code around, the focus of this paper will be Ean-13 numbers. The Ean-13 system is used to represent UPCs: the Universal Product Code, which is used to identify a variety of items, including food, merchandise, and coupons. We have chosen to focus on Ean-13 numbers because they are well defined, and because of their use in merchandise businesses. The principles of our implementation can be easily applied to other bar code systems.

Ean-13 numbers are made up of 12 digits. The digits are broken into four parts (see Figure 1). The first digit represents the type of object this number belongs to. For example, most merchandise begin 0. In the case of merchandise, the next five digits represent the manufacturer of the product. The following five digits represent the individual product from that manufacturer. The final digit is a check digit which is computed as follows.

$$T = 2\left(\sum_{i=0}^{5} d_{2i}\right) + \left(\sum_{i=0}^{10} d_i\right)$$
$$C = 10 - (T \bmod 10)$$

where d_i is the ith digit in the bar code with d_0 as the first digit, and C is the check digit. [1]

These digits are encoded into 15 characters; each one made of seven units of space, and each containing two variable width bars. The bar code is composed of a start guard, followed by six left characters, a middle guard, six right characters, and finally an end guard. The digits are encoded using the data from Figure 2. Left characters are encoded with the negation of right characters. This allows us to encode the direction of the bar code by using the parity of the number of spaces which the black bars take up. Generally, the number which an encoding represents is written below the bar code, usually with the first and last digits written to the sides, though this is

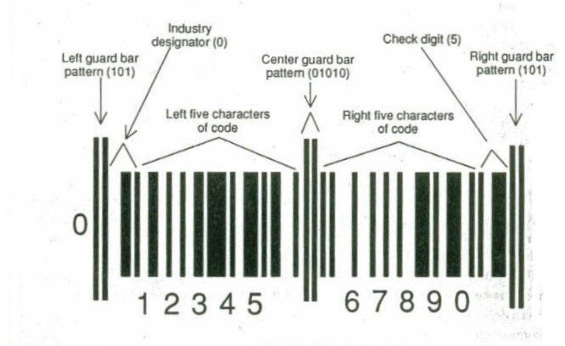

Figure 1: Structure of an EAN-13 Bar Code [1]

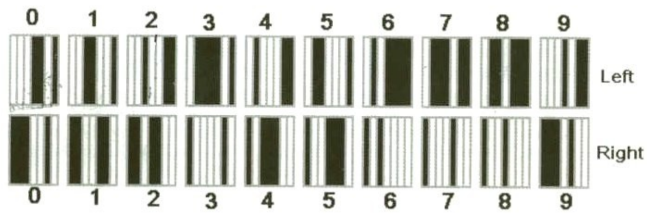

Figure 2: Ean-13 Characters [2]

not a requirement. [1 and 2]

The most widespread method of interpreting bar codes is to use a one-dimensional scanning system. Such systems include light pens which read intensity values as they are passed over an image, linear scanners which are automated versions of the pen scanner, CCD-based systems which use a linear light source, and a one dimensional CCD to collect data, and multi-angle systems which attempt to scan the bar code at several angles. There are some commercially available two-dimensional solutions as well. These solutions cost anywhere from under $100 to several thousands of dollars. [3]

Previous applications of computer vision for interpreting bar codes have generally been focused at interpretation, and not at tracking. Interpretation is subdivided into two subsections: scanning and decoding. Scanning focuses on methods of generating a waveform from the bar code. Decoding focuses on translating the waveform into a set of numbers.

While trying to reduce the error in scanning prescriptions at a pharmacy distribution center, Ruben Muniz, Lois Junco, and Adolpho Otero proposed using a Hough transformation in order to remove extraneous marks. They were able to increase the success rate of reading bar codes which were partially obscured by doctors signatures from 50% success to 85% success. [Muniz] We will use a form of this later in this project.

In a resent article in SIAM News, Todd Wittman proposes a method for improving decoding by attempting to model the noise that prevents a scan from being a square wave. [5] Using a mathematical model for Gaussian noise as applied to a bar code, he proposes to minimize the total-variation energy. By doing this, we can better recover the bar code data. This algorithm is not efficient enough for real time applications, however.

Where most research in bar codes is done on just one portion of reading bar codes, this project is focused on producing a complete implementation of a bar code reader which takes data from a two-dimensional video stream.

Implementation Overview

In order to interpret a bar code, our implementation goes through several phases. First, the frame is imported. A copy of this image is prepared by applying a three by three Gaussian blur convolution to the image in order to reduce any salt-and-pepper noise. This image is sent to the locater. The locater finds a best guess as to the location of a bar code in the image, and determines the probability that what it found is a bar code. This is then tracked to prevent re-processing of already interpreted bar codes in the video stream. If a likely bar code was found which hasn't been interpreted successfully yet, then that image is passed on to a bar code interpreter which will attempt to scan and decode the image. Due to the error detection built into the Ean-13 system, we can be reasonably certain whether or not a number we have read is valid. If the number was read successfully it is marked in the locater and the number is passed to the controlling application (e.g. Cash register).

Locating the Bar Code

In order to interpret the bar code, we must first determine what is bar code and what is background. Because bar codes are made up of many parallel lines, and because the bar code must take up a good portion of the image plane to be valid input for this application, we can find the bar code by finding edge pixels with the mode gradient angle. In order to do this, we find the edges in the image by applying Sobel convolutions to the image, resulting in vertical and horizontal response matrices. These matrices contain information which allows us to determine both the magnitude and the angle of the gradient on all edge pixels in the image. Random noise does not effect the mode angle because, statistically, it will cancel itself out. The mode edge angle is determined by creating a histogram of all angles who's gradient magnitude is significant. The mode of this histogram is found. A binary image is then formed by assigning all significant pixels with the mode angle to true. A centroid is then found by averaging the locations of all responding pixels, and a bounding box is formed. The bounding box, and the centroid location are returned for use in interpreting the image.

This seems to produce simple, high speed results which track a bar code even in a very noisy image.

We will discuss the tracker later, however, it is important to note that if a bar code is located, and determined to be the one that is being tracked, the frame will be dropped at this point, and the next portion of the implementation will not be executed.

Acquiring the Waveform

Once the location of the bar code is determined, the window defined by the bounding box is processed by the bar code reader. The first step in reading the bar code is scanning it. Like scanning in a supermarket. This step of the program converts the image of the bar code into a linear array of gray values. This is done using a variation of the classical hough transformation for finding lines. Every pixel in the sub image votes for every line which passes through it according to it's pixel value. This results in a parameter space filled with sine waves which intersect each other at the angle which is perpendicular to the bars of the bar code (see Figure 3). Because we have determined the mode gradient angle in a previous step, and because the mode gradient angle is similar to the angle perpendicular to the bar code, we can prevent calculating the whole parameter space which is costly in execution time, and can instead calculate only one row of the parameter space. This reduces the scanning algorithm by a factor of 300. The resulting row of the parameter space is a good approximation of a scan of the bar code. Superimposing this row of parameter space onto the original image perpendicular to the bars

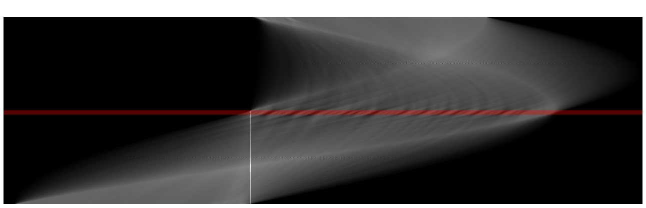

Figure 3: Hough Parameter Space

Figure 4: Bar Code with Paramater Space, and its Second Derivative

in the bar code creates a very close match to the original image (see Figure 4).

Bar code scans are traditionally considered as waveforms which result from reading the reflectivity of the image while scanning perpendicular to the bars. The resulting row of parameter space which we have calculated is a representation of this waveform. This waveform may, however, have an offset due to the extra data caused by an imperfect bounding box. This should not make a difference to the second derivative.

Because this calculation is based on a distance and an angle, there is no difference between scanning a bar code at an angle, and a bar code square to the image plane.

Throughout the rest of this paper, when referring to the waveform, it is this row of the parameter space we are referring to.

Decoding the Bar Code

Once the bar code waveform has been retrieved, we attempt to decode it. There are many methods for decoding this waveform which range in complexity from algorithms that a small circuit chip can process it in real time, to methods that can take several minutes on high-end workstations. [5] For our implementation, we have decided to use the classic second-derivative method. We are given the waveform from the previous step. In a perfect world, this would be a square wave where the low points are bars, and the high points are spaces. Unfortunately our waveform is not square due to static, discretation errors, and ink spread. What we have instead is a rounded waveform

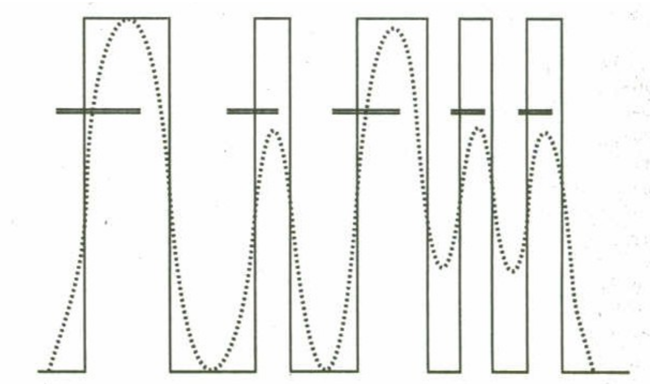

Figure 5: Ideal Waveform and typical rounded scan

which increases on spaces, and decreases on bars. The extrema of this waveform lie on the centers of the bars and spaces, and the inflection points lie on boundaries between the bars and spaces (see Figure 5). To find the inflection points, we take the second derivative and find the zero crossings. This gives us a set of widths of potential bars and spaces. This information is scanned for patterns which are similar to the beginning, middle, and end characters which are matched up to find a space that might contain a valid Ean-13 bar code. We then go through, and break up the data into four-width characters. Two t-values are calculated for each character in the following manner.

Given widths $W_0, W_1, W_2, and\ W_3$:

$$T_1 = W_0 + W_1$$

$$T_2 = W_1 + W_2$$

Each Ean-13 character has a different set of t-values, with the exception of the pairs one and seven, and two and eight. We can distinguish within these pairs with a simple comparison of individual bar widths. [1] We use t-values to significantly reduce the effects of ink-spread which is a result of the physical properties of the ink and medium causing the bars to widen.

Tracking

A simple method of tracking is implemented which takes into account the distance that the centroid of the bar code has moved in one frame. A constant is set which determines the maximum distance the centroid can move before it is considered a new bar code. This serves well because it is extremely fast to calculate, and appears to track the bar code successfully. If the previous step indicates that the bar code was successfully interpreted, then the tracker is marked so that it is not read again.

Uniqueness of Design

It is hard to judge the uniqueness of this design because most solutions for reading bar codes are created in industry where they are kept as proprietary designs. There have been a few commercial solutions which read bar codes using a 2-dimensional ccd, however we cannot compare the uniqueness of our solution to these because they are closed, proprietary solutions. Also Because the restrictions required for using linear scanners are conducive to the standard uses for bar codes, development on other systems for interpretation have been slow. We were not able to acquire a copy of the paper written by Muniz on using the hough transformation, so we cannot make a full comparison to that paper. [4] However we infer from the abstract that this method is similar to ours in some respects. No project which we have found has integrated the steps of bar code tracking and interpretation to this degree, using the results from one stage to increase efficiency in another.

Results and Contributions

The waveform which we acquire is quite good on most images. The decoding scheme we use, however, is not tolerent of slight noise in the waveform. Using better schemes of edge detection, we could improve the scanning accuracy dramatically. The waveform we generate is what standard 1-dimensional decoders use, so this system could easily be used in conjunction with commercial decoders.

The waveform we generate is angle-independent, and resistant to noise, and damage. Figure 6 shows a bar code at different angles, and their waveforms superimposed over them.

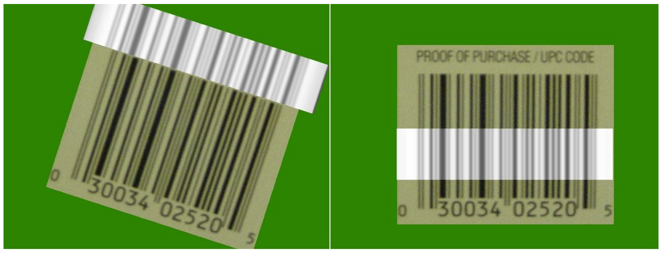

Figure 6: Scan on angles

Because the Sobel mask only provides an approximation of the angle perpendicular to the bar code, the more off-square the image is, the wider the spread the result is. This could be predicted, and corrected, however it is generaly unnessasary since EAN-13 numbers are resistant to scale.

Along with angle independence, this system is very robust against noise, and damage. Figure **7** shows a bar code which has been deactivated in a common manner: putting a line through the bar code so that it runs from top to bottom makes it impossible to find a line which can be read. Because our system uses the entire bar code to create the waveform, we can still read them. It is useful to be able to deactivate bar codes, and so it seems that this is not a good feature. However, there are many situations where the bar code can become unreadable on any scan—such as dirt, or other marks—where this is a desirable result.

This algorithm does not run in real time—that is it does not run at 30 frames per second—however, it does run at 2 frames per second even with several instances of debug information which take a considerable amount of time copying information, and accessing the hard drive. We believe that with a minimum of time spent, we could make this algorithm run at over 5 frames per second, which would be fast enough for an application in supermarkets, and other scanning points.

This system is not, by itself, good at interpreting bar codes, however it is quite good at tracking, and scanning bar codes. We believe that this algorithm will be quite useful when used with a robust decoder, such as those currently used in scanning systems.

References

[1] T. Pavlidis, J. Swartz, and Y. Wang, Fundamentals of Bar Code Information Theory, *Computer* 23, 4. 74-86, 1990.

[2] XlogiX, BarCode Tricks, *2600 The Hacker Quarterly* 21, 1 (Spring 2004), 25-26.

[3] HHP Homepage http://www.hhp.com/

[4] Muniz, R., Junco, L., Otero, A., A Robust Software bar code Reader using the Hough transform. Proceedings of IEEE International Conference on Information Intelligence and Systems. (1999), 121-135.

[5] Wittman, T., Lost in the Supermarket: Decoding Blurry Barcodes. *SIAM News* 37, 7 (September 2004).

[6] Horowitz S., A Syntactic Algorithm for peak Detection in Waveforms with Applications to Cardiography. *Communications of the ACM* 18, 5 (May 1975), 281-285.

[7] Joeseph, E., Pavlids, T., Bar Code Waveform Recognition Using Peak Locations. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16, 6 (June 1994) 630-640.

[8] Hudson, P., Game programming. *Linux Format* 57 (September 2004), 78-81.

[9] Shapiro, L. Stockman, G., Computer Vision. 2001.