Published in final edited form as:
Proc Can Conf Comput Robot Vis. 2009 May 25; 2009: 61–67. doi:10.1109/CRV.2009.31.

A Bayesian Algorithm for Reading 1D Barcodes

Ender Tekin and
The Smith-Kettlewell Eye Research Institute

James Coughlan
The Smith-Kettlewell Eye Research Institute

Ender Tekin: ender@ski.org; James Coughlan: coughlan@ski.org

Abstract

The 1D barcode is a ubiquitous labeling technology, with symbologies such as UPC used to label approximately 99% of all packaged goods in the US. It would be very convenient for consumers to be able to read these barcodes using portable cameras (e.g. mobile phones), but the limited quality and resolution of images taken by these cameras often make it difficult to read the barcodes accurately. We propose a Bayesian framework for reading 1D barcodes that models the shape and appearance of barcodes, allowing for geometric distortions and image noise, and exploiting the redundant information contained in the parity digit. An important feature of our framework is that it doesn't require that every barcode edge be detected in the image. Experiments on a publicly available dataset of barcode images explore the range of images that are readable, and comparisons with two commercial readers demonstrate the superior performance of our algorithm.

1. Introduction

The 1D barcode was developed as a package label that could be swiftly and accurately read by a laser scanner. It has become ubiquitous, with symbologies such as UPC used to label approximately 99% of all packaged goods in the US [1]. There is a strong demand for systems to read 1D barcodes from images acquired by portable cameras (such as on cell phones), but the 1D patterns are often poorly resolved by these cameras because of motion blur and an inability to focus sufficiently close. A variety of 2D barcodes have been designed that are much better suited for camera acquisition [2,3], but for the next several years 1D barcodes will remain the dominant type of label for most packaged goods.

There has been a variety of research done on algorithms for reading 1D barcodes, which we survey briefly here. Most past work has dealt with scanline data (typically acquired by a laser scanner), using waveform analysis, deblurring and other signal processing techniques to detect edges [4,5,6]. The approach in [7] is similar to ours in its use of probabilistic HMM modeling, but it assumes that every edge in the barcode is detected in the scanline data, which is an assumption that we remove in our algorithm. (The HMM approach assigns edge transition states to the entire sequence of observed edges in a scanline, whereas our approach is a deformable template that uses hidden variables to model the locations of each barcode edge, regardless of which edges are visible.) Another difference is that our approach jointly performs decoding and error correction. There has also been work on reading 1D barcodes from images rather than laser scanner data, some of which uses 2D analysis such as the Hough transform to detect edges [8].

Some recent work reports impressive accuracy [9,10], but the image datasets on which the algorithm were tested are not yet available ([10] states that the data will be available to the public but this is not the case at the time of writing). However, we emphasize that the readability of a barcode image varies greatly depending on the image quality; under some conditions (e.g.

using a camera phone, which has limited resolution and ability to focus close-up) many images are blurry even when the photographer takes the images carefully. For instance, in [9] the barcodes are photographed at a resolution such that the module (width of the narrowest bar) is about 4.5 pixels wide, which may be too high a resolution to achieve consistently with a camera phone. Therefore, it is crucial that the images used to evaluate a barcode reading algorithm be made publicly available.

Our contribution is a Bayesian deformable template model for reading barcodes that is robust to noise due to non-uniform lighting, geometric and perspective distortion and to missing edges and low resolution images. We have posted our dataset online for public use, and plan to add to the dataset as the project continues.

We have not yet tackled the problem of localizing barcodes in cluttered images, but there is work on this problem [11,9] that we will draw on in the future.

2. Bayesian Model

We have devised a Bayesian deformable template [12] model of the barcode that combines prior knowledge of barcode geometry, including the allowed configuration of bars and allowing for geometric distortions, with evidence for edges based on intensity gradients. Our model is also strengthened by exploiting the checksum information embedded in the barcode, which constrains the values of the encoded digits, thereby allowing us to detect *and correct* single-digit errors. In this paper we specialize to a particular symbology that is commonly used in North America, the UPC-A, but we emphasize that our approach will generalize straightforwardly to any 1D barcode pattern.

2.1. Barcode Structure

The UPC-A barcode (see Fig. 1) encodes a string of twelve decimal digits (each digit is an integer from 0 through 9), where the last digit is a checksum that is a function of the previous eleven. The barcode pattern consists of a sequence of black bars and white gaps between the bars (we will refer to both as bars). There are 29 white and 30 black bars, giving a total of $N = 60$ edges. The edges have alternating polarity, and from left to right the polarity of edge i (where $i = 1, \ldots, N$) is $(-1)^i$. Each bar has one of four possible widths: Δ, 2Δ, 3Δ or 4Δ, where Δ is the *modulus* or *fundamental width* of the barcode.

Each digit value is encoded by a sequence of four bars (with total width 7Δ). In addition to the twelve digit regions, there are three ‘guard’ regions in the barcode: the start region (two black bars separated by a white bar) on the left, the middle region separating the sixth and seventh digits (three white bars separated by two black bars), and the end region on the right (the same pattern as the start region). See Fig. 2 for an illustration of these regions. The total width of the barcode is 95Δ.

The 60 edges in the barcode are classified into two categories, fixed and variable. Fixed edges are those associated with the guard bands (also shown in Fig. 2), whose location is independent of the encoded digits. Variable edges are the edges in the digit regions whose locations define the digit encoding.

2.2. Model Basics

In this paper we assume that the bar code has been segmented from the image, and that we know the approximate orientation, which allows us to construct several scanlines across the barcode. We first describe the model for a single scanline cutting from left to right across the barcode; later we will extend the model to multiple scanlines. In addition, in later sections we will describe two variants of the basic model, Model 1 and 2.

The scanline defines an x coordinate system, and the intensity along the scanline is denoted $I(x)$. The edge strength $e(x)$ is defined as the intensity derivative dI/dx. Local maxima and minima of $e(x)$ define the edge locations in the scanline, and the first and last observed edge locations spanning the entire barcode (of width 95Δ) are used to estimate Δ for the scanline. We denote all the information in the scanline by S.

The locations of all $N = 60$ edges are denoted by the sequence $X = (x_1, x_2, \ldots, x_N)$. We denote the fixed edges in X by X_f, and the variable edges by X_v, so that with a slight abuse of notation we can write $X = (X_f, X_v)$.

The unknown digits are denoted by the sequence $D = (d_1, d_2, \ldots, d_{12})$, where $d_i \in \{0, 1, \ldots, 9\}$.

The basic model, $P(X, D|S)$, can be described as a factor graph (see Fig. 3) of the following form:

$$P(X, D|S) = \frac{1}{Z} e^{-L(X,S)-G(X,D)} \tag{1}$$

where $L(X, S)$ is the (log) likelihood term that rewards edge locations lying on high-gradient parts of the scanline, and $G(X, D)$ is the geometric term that enforces the spatial relationships among different edges given the digit sequence.

First we describe the likelihood term $L(X, S)$ in more detail. It is defined as:

$$L(X, S) = \sum_{i=1}^{N} L_i(x_i, S) \tag{2}$$

where the precise form of L_i differs between Model 1 and Model 2 (see details in the next subsections). In both cases it enforces the *polarity constraint* that edge x_i must have polarity $(-1)^i$, i.e. $L_i(x_i, S) = \infty$ if $dI/dx(x_i)$ has the wrong sign.

Note that we can rewrite $L(X, S)$ as two terms, one containing fixed edges and the other containing variable edges:

$$L(X, S) = L_f(X_f, S) + L_v(X_v, S) \tag{3}$$

Next we describe the geometric prior term $G(X, D)$:

$$G(X, D) = G_f(X_f) + G_v(X_f, X_v, D) \tag{4}$$

The first term, $G_f(X_f)$, enforces the appropriate spacing between fixed edges: for instance, we expect that $x_2 - x_1 \approx \Delta$. This expected separation distance is expressed with a quadratic energy term $G_f^1(x_1, x_2) = \beta_f (x_2 - x_1 - \Delta)^2 H(x_2 - x_1)$ for the first two edges, and similarly for all other consecutive fixed edges. Here $H(x)$ is a function that equals one for positive values of x and ∞ otherwise, which enforces the fact that consecutive edges are ordered from left to right instead of right to left; β_f is a positive constant. In addition, since each digit region encompasses a width of 7Δ, we enforce this property with the following energy term: $G_f^4(x_4, x_8) = \beta_f (x_8 - x_4 - 7\Delta)^2 H(x_8 - x_4)$ for the first digit, and similarly for the other digits.

The second term, $G_v(X_f, X_v, D)$, enforces the appropriate spacing between variable edge locations that encode digit values. Each digit value (0–9) corresponds to a sequence of four widths. In order from 0 through 9, the associated width sequences are: (3, 2, 1, 1), (2, 2, 2, 1), (2, 1, 2, 2), (1, 4, 1, 1), (1, 1, 3, 2), (1, 2, 3, 1), (1, 1, 1, 4), (1, 3, 1, 2), (1, 2, 1, 3), (3, 1, 1, 2). These spacings are enforced by energy functions such as $G_v^{1,1}(x_4, x_5, d_1)=\beta_v(x_5 - x_4 - w_1(d_1))^2 H(x_5 - x_4)$, where $w_1(d_1)$ is the first width corresponding to digit d_1 (and β_v is a positive constant).

The next two subsections describe two variants of this model that we have implemented.

2.3. Model 1

Model 1 is the simpler of the two models. In this model, we first normalize $e(x) = dI/dx$ by rescaling it so that its absolute value has a maximum equal to one. We then extract edges as the local minima and maxima of $e(x)$, and denote the sequence as $Y = (y_1, y_2, \ldots y_M)$. The locations y_i are estimated to sub-pixel accuracy by modeling the value of $e(x)$ as locally quadratic. This sequence defines the set of allowed edges ($M \geq N$, meaning that we have observed at least as many edges as needed in the barcode), i.e. allowed values of the x_i. We then define the evidence for each edge as $L_i(x_i, S) = \alpha|e(x_i)|H(e(x_i)(-1)^i)$, which enforces the need for x_i to have the correct polarity.

In Model 1 we estimate the edge locations using the following procedure. First we use the Viterbi algorithm [13] to calculate the fixed edge locations:

$$X_f^* = \arg\min_{X_f} L_f(X_f, S) + G_f(X_f) \tag{5}$$

Then using this result, Viterbi is used again to calcuate the variable edge locations:

$$X_v^* = \arg\min_{X_v} L_v(X_v, S) + G_v(X_f^*, X_v, D) \tag{6}$$

We now have estimated all edge locations X^* before estimating the digit values. (Model 2 *marginalizes* over edge locations instead, which confers certain advantages and disadvantages relative to Model 1.)

To estimate the marginal probability of any given digit we use the following expression:

$$P(d_i|S) = \frac{1}{Z'} e^{-E_i(X^*, D, S)} \tag{7}$$

where D on the RHS has the value of d_i specified on the LHS, and i specifies which digit in the string is to be estimated. Here $E_i(X, D, S)$ comprises all terms in $L(X, S)$ and $G(X, D)$ that depend on the edge locations defining d_i.

If multiple scanlines are used, then we simply average over the marginal probabilities from each scanline to arrive at an overall marginal probability.

2.4. Model 2

Model 2 has the same structure as Model 1. The most important difference, however, is that we allow edge locations X to assume any values on a pixel lattice rather than restricting ourselves to local peaks in $e(x) = dI/dx$. This was done because in noisy images the true location

of an edge may not be associated with a local peak, i.e. *we do not assume that we are able to detect every barcode edge*. We define a pixel quantization on each scanline such that the estimated fundamental width corresponds to approximately 5 pixels, and estimate $e(x)$ on this lattice using linear interpolation.

Since we are no longer restricting ourselves to locations of peak values in dI/dx, we also use the second derivative $e_2(x) = d^2I/dx^2$ to reward locations that are close to local maxima or minima. (This is normalized in the same way that $e(x)$ is normalized.) Then we define the likelihood for an edge as $L_i(x_i, S) = -\gamma\log(|e(x)|(1 - |e_2(x)|)H(e(x_i)(-1)^i)$, which rewards edge strength as well as proximity to a peak in edge magnitude (since $e_2(x)$ attains 0 at a local maximum or minimum). (Here γ is a positive constant.) Again, the $H(.)$ factor forces the gradient at x_i to have the correct polarity.

The second difference with Model 1 is that Model 2 uses factor graph belief propagation (BP) [14] to estimate marginal probabilities of each digit, instead of greedily estimating the most likely edge locations and then estimating digits based on these locations. The factors in Model 2 are defined by exponentiating the energy functions in Sec. 2.2, and so BP estimates the marginal probabilities of each variable in the model, including $P(d_i|S)$.

2.5. Using the checksum

The digits in the barcode digit sequence are not independent but are constrained to obey the following parity equation:

$$\left[3\sum_{i\,\text{odd}} d_i + \sum_{i\,\text{even}} d_i\right] \text{mod} 10 = 0 \tag{8}$$

Thus the first eleven digits are chosen independently and the twelth digit is a parity check digit chosen to satisfy the above constraint. This mechanism allows for verification of the barcode by the reader.

In order to find the most likely digit sequence that is consistent with the data *and* satisfies the parity constraint, we create a new random variable sequence $c_i \in \{0, \ldots, 9\}$ where $i = 1, \ldots, 12$, corresponding to a running parity digit. Let

$$c_i = \begin{cases} 3d_1 \ \text{mod} 10, & \text{if } i=1 \\ (3d_i + c_{i-1}) \ \text{mod} \ 10, & \text{if } i \text{ is odd} \\ (d_i + c_{i-1}) \ \text{mod} \ 10, & \text{if } i \text{ is even} \end{cases} \tag{9}$$

Observe that

$$P(c_1, c_2, \ldots, c_{12}) = P(c_1)P(c_2|c_1)\cdots P(c_{12}|c_{11}) \tag{10}$$

Here we are using the marginal probabilities $P(d_i|S)$ (Eq. 7) to define the associated probabilities on the c_i variables (where we omit the conditioning on one or more scanlines for brevity). (For instance, since c_1 and c_2 jointly determine the value of d_2, $P(c_2|c_1)$ is determined by $P(d_2|S)$.) Since the sequence $c_1, c_2, \ldots, c_{12}$ forms a Markov chain, we can use Viterbi (after taking a log to transform to the energy domain) to find the most likely sequence of c_i, and therefore the most likely sequence of d_i. In addition, a multipath Viterbi algorithm is also used

to calculate the second most likely sequence, which is used to evaluate the confidence of the top estimate (see next section).

2.6. Model selection

In our experiments we have found that Model 1 is better for clean images and Model 2 is better for noisier images. We have not yet investigated the possibility of a Bayesian model selection procedure, but have instead devised a simple algorithm to decide which model is appropriate for a given barcode image.

Our algorithm is the following. Model 1 is first run using a single scanline. A confidence criterion (described below) is calculated, and if it is too low then another scanline is used (up to a total of 15 scanlines). If one or more scan-lines have been used, then the overall probabilities for each barcode digit are calculated by simply averaging over the scanlines.

If the confidence criterion is satisfied using Model 1 then the algorithm is done. Otherwise we proceed to run Model 2, again starting with one scanline. More scanlines are evaluated as needed (up to a total of 15 scanlines considered individually, i.e. without aggregating multiple scanlines or averaging digit probabilities), and if the confidence criterion is satisfied for any scanline then the algorithm is done. Otherwise the algorithm reports failure.

The confidence criterion assesses how much more likely the top estimated digit sequence is than other sequences. Specifically, the confidence criterion is fulfilled only if two tests are satisfied. In the first test, the top two digit sequence estimates from the checksum procedure are compared, and if the ratio of the probability of the top sequence divided by the probability of the second most likely sequence is too small then the test fails. (This test assesses the margin of confidence in the top solution.) If this first test succeeds, then a second test is performed. An alternative digit sequence is calculated in which each digit is chosen to be the one that maximizes the marginal digit probability from the checksum procedure. If the alternative digit sequence differs from the most likely sequence by at most one digit, then the second test succeeds and the overall confidence criterion is satisfied.

3. Experimental Results

Models 1 and 2 were implemented in unoptimized Mat-lab code, taking on the order of a few tens of seconds (Model 1) to many tens of seconds (Model 2) to execute per image. We note that a single slice with Model 1 achieves a performance close to the best performance of the joint model, but at higher speed.

We acquired 79 images of UPC-A product barcodes, some photographed using the Nokia N95 camera phone, others using the Nikon Coolpix 4300 camera, and others taken from the internet. All images were saved as jpegs. For each image, the barcode was manually segmented (and oriented roughly horizontally). We first divided our database subjectively into two classes that we dubbed *clean* (see Fig. 4) and *hard* (see Fig. 5) with the expectation that the performance on the clean set should be very good. (We chose this subjective criterion in the absence of any objective criteria in the literature on reading barcodes.) The clean set consisted of 44 images of varying resolutions, distortions and contrast, but otherwise seemed recognizable. The hard set consisted of 35 images that were blurry or distorted, and hence were likely candidates for failure. We have posted our ground truth-annotated database on our website [15], and we will add to it as the project progresses.

From the clean set, we randomly chose 17 images that we used as a training set. Appropriate values of Model 1 and Model 2 parameters (α, β_f, β_v and γ) were estimated by exhaustive search, in which the error rate of the algorithm was evaluated for each combination of parameter values,

and we chose the parameters with highest confidence margins that resulted in all 17 barcodes being read correctly.

We then used our models to try to read the whole set of 79 images. We also compared our algorithm with two commercial barcode readers, Barcode Decoding Software from DataSymbol [16] and bcTester Barcode Recognition from QualitySoft [17], on the same set of images. Our results are summmarized in Table 1, and detailed results are tagged in [15]. We stress that when comparing our results to the commercial software, we adjusted the options to make sure that the software was only trying to recognize a UPC-A barcode so as to increase the accuracy (and to permit a fair comparison with our algorithm, which at this time assumes the UPC-A structure). (Both commercial algorithms automatically segment the barcode from the image, whereas our algorithm assumes the barcode has already been roughly segmented; the identical segmented images were given to all three algorithms.) Our algorithm succeeded on all images that were correctly read by the commercial readers, and it also succeeded on other images. Like the commercial readers, our algorithm always reported failure when it was unable to estimate the bar code correctly.

Some examples of clean images are shown in Fig. 4, and of hard images in Fig. 5. Note that, while most of the edges in the clean images are easy to extract, some edges are fuzzy, and localization noise means that it is sometimes difficult to estimate bar widths correctly. In the hard images, many of the edges are very difficult to resolve (often because of low contrast and motion blur), and in some cases (top of Fig. 5) the modulus is quite narrow, approximately 2 pixels wide. The examples in Fig. 5 give an idea of the limits of our current algorithm.

While Model 2 is better able to cope with faint or missing edges than Model 1, its ability to choose edges that don’t lie on image gradient peaks adds uncertainty that sometimes compromises its ability to infer the correct digit sequence. Thus, we have found that Model 1 performance is superior to that of Model 2 on cleaner images, but that Model 2 is necessary for hard images (for which Model 1 may have difficulty finding good edges even if they are all visible). By adding additional cues, and tuning the model parameters more carefully (and on a larger training set), we hope that in the future Model 2 will improve to the point where it will completely replace Model 1. We discuss some possible improvements in the next section.

4. Discussion

We have described a novel Bayesian algorithm for reading 1D barcodes in noisy images. The algorithm is based on a deformable template that is robust to geometric deformations as well as image noise, and uses the checksum digit both to improve its chances of estimating the correct digit sequence and to rule out impossible sequences. An important feature of our framework is that it doesn’t require that every barcode edge be detected in the image. Experiments on a publicly available dataset of barcode images explore the range of images that are readable, and comparisons with two commercial readers demonstrate the superior performance of our algorithm.

In the future we plan to improve the performance of the algorithm by using a more principled learning procedure from a larger training set, such as conditional random fields (CRFs) [18], and to test its performance on a substantially larger test set. Such a procedure will also allow us to incorporate multiple cues, such as the scarcity of edges *inside* the bars, in addition to the presence of edges at the bar boundaries. We hope to integrate the entire model into a single factor graph that seamlessly combines multiple scan lines and the checksum information. We are currently investigating the possibility of explicitly modeling image blur (due to poor focus and camera motion), which is a major source of image degradation. It may also be possible to

augment the barcode model with a simple OCR digit module that reads the digits printed below the barcode.

Finally, we will extend our model to read other common barcode patterns (e.g. UPC-E and EAN-13), and use a simple model selection procedure to decide which barcode model is appropriate for each image. Then we will implement a barcode localization algorithm to automatically find the barcode boundaries in a cluttered image, and write a fast version of the entire algorithm in C++. Ultimately we hope to port the algorithm to the camera phone platform in a form that blind and visually impaired persons can use to identify packaged goods.

Acknowledgments

The authors were supported by The National Institute of Health grant 1 R01 EY018890-01 and The National Institute on Disability and Rehabilitation Research (NIDRR) grant H133G030080.

References

1. Garg, V.; Jones, C.; Sheedy, C. 17 billion reasons to say thanks: The 25th anniversary of the U.P.C. and its impact on the grocery industry. 1999. [Online]. Available: http://barcodes.gs1us.org/dnn_bcec/Documents/tabid/136/EntryId/34/DMXModule/731/Default.aspx
2. Wang, H.; Zou, Y. Camera readable 2d bar codes design and decoding for mobile phones. 2006 IEEE International Conference on Image Processing; 2006. p. 469-472.
3. Parikh, D.; Jancke, G. Localization and segmentation of a 2d high capacity color barcode. IEEE Workshop on Applications of Computer Vision; 2008. p. 1-6.
4. Joseph E, Pavlidis T. Bar code waveform recognition using peak locations. IEEE Transactions on Pattern Analysis and Machine Intelligence 1994;16(6):630–640.
5. Shellhammer S, Goren D, Pavlidis T. Novel signal-processing techniques in barcode scanning. IEEE Robotics and Automation Magazine 1999;6:57–65.
6. Joseph E, Pavlidis T. Deblurring of bilevel waveforms. IEEE Transactions on Image Processing 1993;2
7. Krešić-Jurić S, Madej D, Santosa F. Applications of hidden markov models in bar code decoding. Pattern Recogn Lett 2006;27(14):1665–1672.
8. Muniz, R.; Junco, L.; Otero, A. A robust software barcode reader using the hough transform. International Conference on Information Intelligence and Systems; 1999. p. 313-319.
9. Wang K, Zou Y, Wang H. 1d barcode reading on camera phones. International Journal of Image and Graphics 2007;7(3):529–550.
10. Wachenfeld, S.; Terlunen, S.; Jiang, X. Robust recognition of 1-d barcodes using camera phones. 9th International Conference on Pattern Recognition; 2008.
11. Chai, D.; Hock, F. Locating and decoding EAN-13 barcodes from images captured by digital cameras. Information, Communications and Signal Processing, 2005 Fifth International Conference on; 2005. p. 1595-1599.[Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1689328
12. Yuille A. Deformable templates for face recognition. Journal of Cognitive Neuroscience 1991;3
13. Rabiner LR. A tutorial on hidden markov models and selected applications in speech recognition 1990:267–296.
14. Kschischang FR, Frey BJ, Andrea Loeliger H. Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory 2001;47:498–519.
15. Barcode images database. 2009. [Online]. Available: http://www.ski.org/Rehab/Coughlan_lab/Barcode/
16. DataSymbol. Barcode decoding software. [Online]. Available: http://www.datasymbol.com/barcode-recognition-sdk/barcode-reader/download-barcode-reader-activex.html
17. QualitySoft. bcTester barcode recognition. [Online]. Available: http://www.qualitysoft.de/indexen.html

18. Lafferty, J.; McCallum, A.; Pereira, F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. Proc. 18th International Conf. on Machine Learning; Morgan Kaufmann, San Francisco, CA. 2001. p. 282-289.[Online]. Available: http://citeseer.ist.psu.edu/lafferty01conditional.html

Figure 1.
UPC-A barcode, encoding 12 digits.

Figure 2.
Fixed edges of UPC-A barcode shown as dashed red lines. Labels on bottom denote the guard regions and the 12 digit regions between the fixed edges.

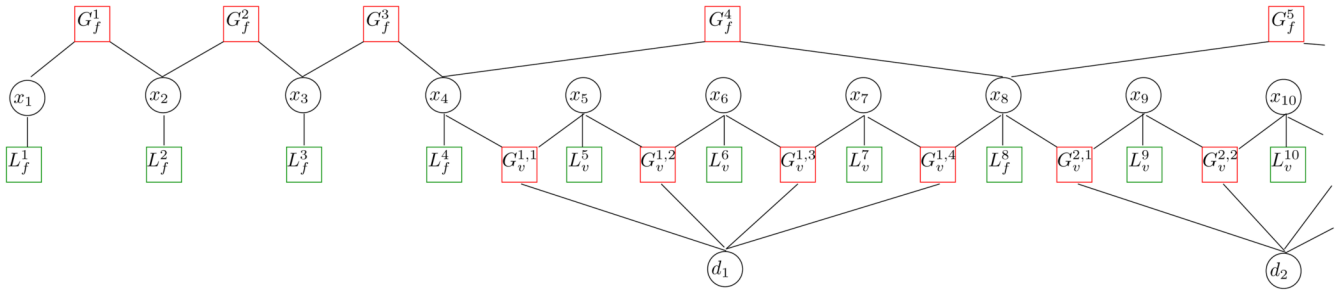

Figure 3.
Factor graph used in Model 1 and 2. Variables are drawn as circles and factors (interactions among variables) as boxes. Factors enforcing geometric constraints are drawn in red and factors incorporating scanline edge information are drawn in green.

Figure 4.
Examples of clean images from our dataset. Top: read correctly by our algorithm and both commercial readers; middle: read correctly by our algorithm and Barcode Decoding Software; bottom: read correctly only by our algorithm.

Figure 5.
Examples of hard images from our dataset. Top: read correctly only by our algorithm (with modulus less than 2 pixels); middle and bottom: read correctly by no algorithm.

Table 1

Number of barcodes correctly read.

	Our Model	Barcode Decoding	bcTester
Learn	17/17	17/17	4/17
Clean	42/44	39/44	13/44
Hard	2/35	0/35	0/35