2006

Locating 1-D Bar Codes in DCT-Domain

Alexander Tropf
Edith Cowan University

Douglas Chai
Edith Cowan University

LOCATING 1-D BAR CODES IN DCT-DOMAIN

Alexander Tropf and Douglas Chai

Visual Information Processing Research Group
School of Engineering and Mathematics, Edith Cowan University
100 Joondalup Drive, Joondalup WA 6027, Perth, Australia

ABSTRACT

Today's digital cameras and camera phones allow users to capture bar codes, which are used to uniquely identify consumer products. In this paper a fast algorithm is proposed that locates a 1-D bar code in the DCT-domain of a bar code image taken by a digital camera. The algorithm uses the DCT-transform properties to distinguish bar code from other texture, morphological operations to smooth the detected bar code area and the features of the extracted area to detect position and orientation of a bar code in the image.

1. INTRODUCTION

In today's modern society, almost every consumer product has a unique 1-D bar code for identification. With the use of a bar code laser scanner, information about a product such as description and price can be easily accessed.

To enable consumer to obtain information about a consumer product at home or in a supermarket, a scanning device that can decode the product's bar code, and a communication device that retrieves the information from a consumer product server, are required. Today's camera phones can handle both tasks: they can take a picture of the bar code and connect to a consumer product server to obtain information about the product.

The critical part of enabling a camera phone for bar code reading is that it first has to locate the bar code in the image and then it requires to decode the bar code. In this article we shall concentrate on the first part: locating the bar code.

Bar code localisation in computer images is not new. There exist numerous approaches for 1-D bar code location using analysis in the spatial domain [1–5], Gabor filtering [6] and analysis in the Wavelet domain [7]. However using the DCT-domain for bar code location has not been investigated yet.

JPEG which uses DCT prior to data quantisation is one of the most common compression standards for digital images. Because of this fact, JPEG codecs are usually implemented in a camera phone's hardware [8, 9] which will provide fast access to the DCT-domain of a bar code image.

This paper is organised as follows: Section 2 investigates the properties of 1-D bar codes in the DCT-domain. Our algorithm is presented in Section 3. Experimental results are given in Section 4, while concluding remarks can be found in Section 5.

2. PROPERTIES OF 1-D BAR CODES IN THE DCT-DOMAIN

Basically, JPEG compression involves the following steps:

- DCT is performed on each 8×8 block of the image.
- The DCT coefficients are quantised.
- The quantised DCT coefficients are encoded with Huffman coding to generate the JPEG bit-stream.

The DCT transforms an 8×8 image block into 64 DCT coefficients. The first DCT coefficient is the "DC-value" since it gives the average value of the image block. The remaining coefficients are the "AC-values" that give the spatial frequencies of an image block in ascending order.

A 1-D bar code can simply be described as an alternating sequence of black and white stripes in one specific direction. Consider two simple cases:

1. The bar code is aligned horizontally and the black and white stripes alternate in x-direction.
2. The bar code is aligned vertically and the black and white stripes alternate in y-direction.

Let us have a look at the DCT-coefficient in the bar code region of the first case. There should be AC-coefficients in x-direction of high magnitude, whereas the coefficients that represent alternating y-components should be ideally zero, or at least very small. In the second case there will be strong AC-components in y-direction and no or only small components in x-direction. Hence DCT-coefficients of bar code regions not only can be distinguished from non-bar code regions, they also give information about the orientation of the bar code. In case the bar code is aligned under an arbitrary angle, the DCT-coefficients of the bar code area will have AC-components in both directions, where the centre weight of each frequency component should give the approximate angle. For example if the alignment angle is $45°$, the bar code AC-coefficients will have an equal portion in x- and y-direction.

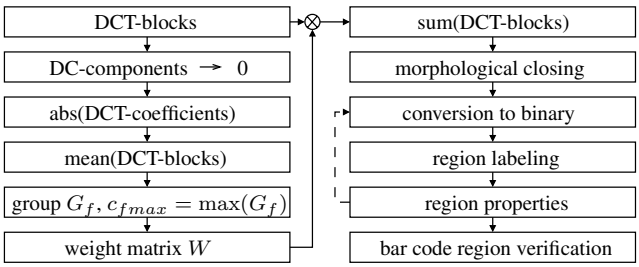

Fig. 1. Flowchart of bar code location algorithm

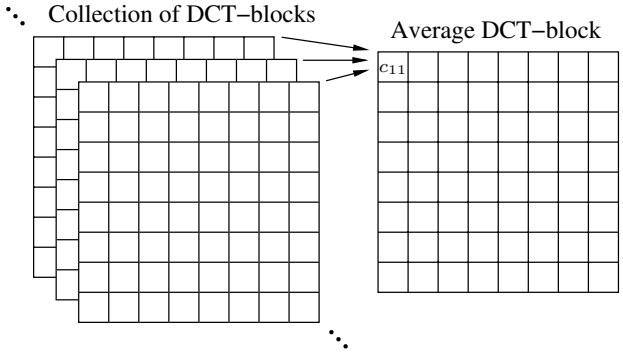

Fig. 2. Computing average DCT-block from all DCT-blocks

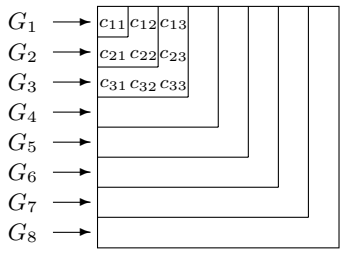

Fig. 3. Grouping of DCT-coefficients from each frequency range

3. BAR CODE LOCATION ALGORITHM

Here, we propose a bar code location algorithm that works on the DCT-coefficients on luminance of a bar code image. To be able to locate bar code DCT-coefficients and to assist in the decoding step, we set the constraint that a bar code should cover an area of at least 10% of an image. The ideal outcome is to detect all lines of the bar code, so that a simple algorithm can be devised for decoding.

A flow chart of our proposed bar code location algorithm is given in Figure 1. The steps of the algorithm are described as follows:

1. Compute magnitude of DCT-coefficients. Only the magnitude of the AC-values is of interest.

2. Set DC-components to 0. Only the AC-values are required to locate the bar code area.

3. Calculate an average DCT-block from all 8×8 blocks of DCT-coefficients (see Figure 2).

4. Take the average DCT-block and group all DCT-coefficients c_{ij} of one frequency range f together into an array G_f, so that $G_f = c_{ij}, (i = f) \wedge j = \{1, \ldots, f\} \vee (j = f) \wedge (i = \{1, \ldots, f-1\})$. For example $G_3 = \{c_{31}, c_{32}, c_{33}, c_{13}, c_{23}\}$ (see Figure 3). Then calculate the largest DCT-coefficient $c_{f\max} = \max(G_f)$, from each frequency range f in the average block. The coefficients c_{fmax} now indicate the coefficients in the bar code area that are strongest.

5. Compute a weight matrix W of dimension 8×8, where each element $w_{i,j}$ is defined as

$$w_{i,j} = \begin{cases} k_e, \text{ if } c_{ij} = c_{f\max} \\ -k_d, \text{ else} \end{cases}$$

where k_e and k_d are the emphasis and deemphasis factors, respectively. A relationship between k_e and k_d will be determined in Section 4, but generally one has to choose $k_e > k_d$ to emphasise bar code coefficients more than deemphasising other coefficients.

6. Perform an element multiplication of each 8×8 DCT block with W. Then calculate the sum of each DCT-block. The higher the sum of a DCT block, the higher is the likelihood that it belongs to the bar code region. The DCT-sums make up a subsampled DCT-image by factor 8 in each dimension, which will increase computation time of subsequent steps. Negative values will be set to 0. Positive values will be scaled to the range [0,255], to construct a gray-scale image.

7. Perform morphological closing on the gray-scale image from Step 6. This will smooth the bar code area. In most cases the bar code area now has the highest intensity compared to surrounding regions.

8. Convert gray scale data into binary using the Otsu thresholding technique [10].

9. Label all 8-connected regions as it is described in [11].

10. At this stage we know that the bar code region has to be large and of approximate rectangular shape. In a simple case the bar code area is much larger than other areas. Therefore the algorithm will choose the largest region as bar code area and proceed with Step 11. In case there are a few large regions that differ in size by factor 2 or less, the one with the highest "solidity" is taken. The term solidity here refers to the proportion of pixel

in the convex hull that are also in the region. If both areas have a high solidity, the bar code detection will go back to Stage 8 and set a high threshold for the binary image conversion. Since the bar code area has a high intensity after Step 7, it will be the first one to show up. Therefore Step 10 chooses the largest area as bar code region and checks the solidity. The algorithm will then repeat Steps 8 to 10 until the result is satisfactory, or abort if no bar code region can be distinguished after a few iterations.

11. In the last stage the bar code region has to be verified to be rectangular to a certain degree. The extrema points [11] are used to check whether the area conforms to a rectangular.

In the end we obtain a binary image mask, which identifies the bar code region of the image.

4. RESULTS

We are using digital bar code images of size 1280×960 pixel to test the algorithm. First of all the emphasis and deemphasis factors have to be obtained. If the ratio k_e/k_d is too low, the bar code area is not emphasised enough and parts of the bar code can be missing. On the other hand, if the ratio is chosen too high, texture is not deemphasised enough and will show up as high intensity area. Experimental results have shown that choosing $k_e/k_d = 3$, gives reasonable results.

Figure 4 shows the result of locating a horizontally aligned bar code. Since the bar code area has only vertical edges, only the DCT-coefficients that represent changes in x-direction will be emphasised. Vertical edges that appear outside of the bar code area are emphasised as well (see upper right corner in Figure 4b). However they can be easily distinguished from the bar code region in Step 10 of the algorithm. The example also shows that texture is deemphasised and it appears only at low intensity after the morphological operation (see left side of Figure 4b). Note that the inclusion or exclusion of numbers beneath the bar code does not have a significant impact of the subsequent decoding process.

Figures 5 and 6 show the results of locating a diagonally and vertically aligned bar code, respectively. It shows that our algorithm is rotation invariant to a high degree.

The detection works even if the bar code has a high content of texture (see Figure 7).

In most cases the bar code was located correctly. However if the percentage of texture in a bar code image is too high compared to the bar code region, the algorithm fails.

5. CONCLUSION

We have shown in this paper that bar code location in the DCT-domain produces good results. Furthermore, the pro-

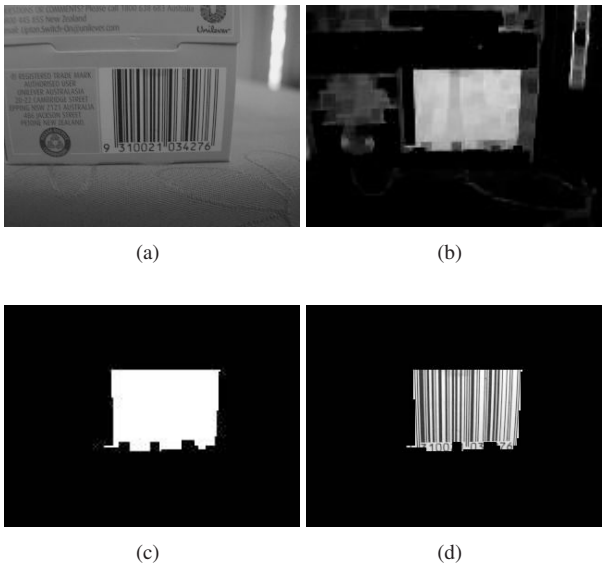

Fig. 4. Result of locating a horizontally aligned bar code: (a) original image; (b) result after morphological operation; (c) binary image mask; (d) located bar code

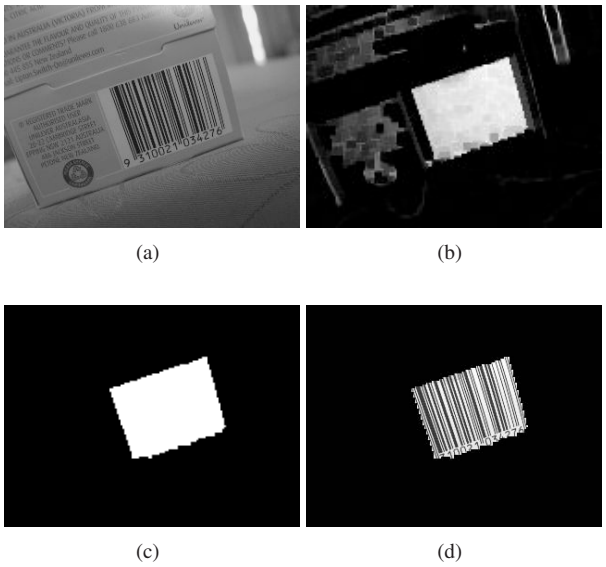

Fig. 5. Result of locating an arbitrarily aligned bar code: (a) original image; (b) result after morphological operation; (c) binary image mask; (d) located bar code

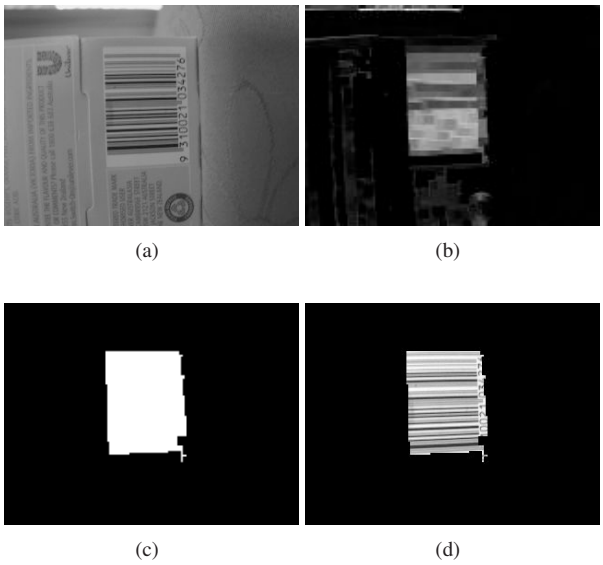

Fig. 6. Result of locating a vertically aligned bar code: (a) original image; (b) result after morphological operation; (c) binary image mask; (d) located bar code

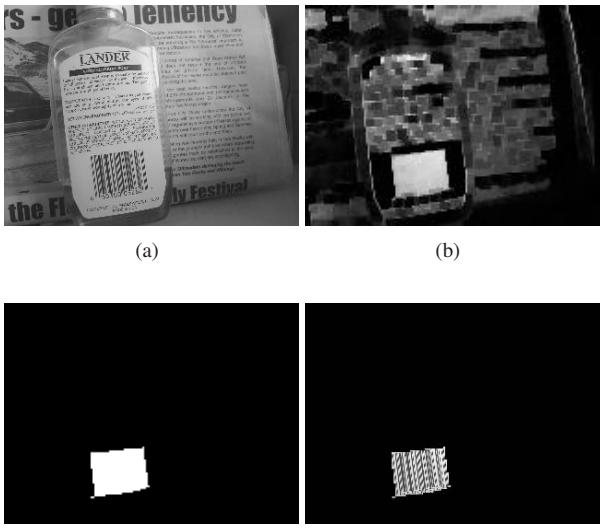

Fig. 7. Result of locating a bar code in image with strong surrounding texture: (a) original image; (b) result after morphological operation; (c) binary image mask; (d) located bar code

posed algorithm is relatively simple to implement and performs very fast. To increase robustness of the algorithm, the bar code region estimation and bar code region verification in the last two steps of our algorithm should be improved. However the algorithm in its current state successfully detects a high number of bar code areas on consumer goods.

The future work will involve an implementation of the bar code location algorithm in JAVA on a camera phone and the implementation of a suitable bar code decoding algorithm.

6. REFERENCES

[1] C. Viard-Gaudin, N. Normand, and D. Barba, "A bar code location algorithm using a two-dimensional approach," in *Proc. IEEE ICDAR'93*, Oct. 1993, pp. 45–48.

[2] S.-J. Liu, H.-Y. Liao, L.-H. Chen, H.-R. Tyan, and J.-W. Hsieh, "Camera-based bar code recognition system using neural net," in *Proc. IEEE IJCNN'93*, vol. 2, Nagoya, Oct. 1993, pp. 1301–1305.

[3] R. Howlett, S. Berthier, and G. Awcock, "Determining the location of industrial bar-codes using neural networks," in *Proc. IEE IPA'97*, vol. 2, July 1997, pp. 511–515.

[4] S. Arnould, G. Awcock, and R. Thomas, "Remote bar-code localisation using mathematical morphology," in *Proc. IEE IPA'93*, vol. 2, July 1999, pp. 642–646.

[5] D. Chai and F. Hock, "Locating and decoding EAN-13 barcodes from images captured by digital cameras," in *Proc. IEEE ICICS'05*, Dec. 2005, pp. 1556–1560.

[6] A. K. Jain and Y. Chen, "Bar code localization using texture analysis," in *Proc. IEEE ICDAR'93*, Oct. 1993, pp. 41–44.

[7] R. Oktem, "Bar code localization in wavelet domain by using binary morphology," in *Proc. IEEE SIU'04*, Apr. 2004, pp. 499–501.

[8] Epson, "New Epson mobile graphics engine for enhanced music playing on mobile phones," June 2005. [Online]. Available: http://www.epson.co.jp/e/newsroom/2005/news_2005_06_27.htm

[9] 3G, "New mobile media processors for mobile phones," Mar. 2004. [Online]. Available: http://www.3g.co.uk/PR/March2004/6886.htm

[10] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybern.*, vol. 9, no. 1, pp. 62–66, 1979.

[11] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*. Addison-Wesley, 1992, vol. 1.