An Algorithm for Visual Code Marker Extraction and Processing

Ahmet Altay, Emre Oto
Department of Electrical Engineering
Stanford University, Stanford, CA 94305

Abstract— **We present the design, implementation and performance evaluation of an algorithm devised for detection, extraction, and processing of visual code markers in images obtained via low-resolution cameras. The algorithm employs a myriad of techniques to locate, extract, and process the visual code markers for embedded binary data. The system performance has been assessed on a set of training images, and the algorithm has been observed to demonstrate high coherence in code marker detection and low bit-error-rate in binary data extraction. The algorithm also exhibits a significantly low execution time, which hints at its feasibility for implementation in low-power mobile devices.**

I. Introduction

Visual code markers have evolved as a bridge connecting the physical world to the cyber-world, as a tool that contributes to stronger human-computer interaction and ubiquitous computing [1]. Significance of code markers is constrained by camera phones. Thus any algorithm destined to process visual code markers must be capable of rapid processing of low-resolution, tilted, cluttered images under power and speed constraints [2].

In this paper, we present an algorithm for processing of low-resolution images acquired by a CCD camera containing visual code markers. The processing of the visual code marker entails the extraction of the binary vector embedded in the pattern. The algorithm takes a color image as input and returns the location of and the binary data in the visual code markers within the image.

The next section describes the morphology of the visual code markers subject to this paper. The mechanics of the proposed algorithm are presented in the following section. The performance of the algorithm on training set images is assessed, and the feasibility of implementation of this algorithm for real world applications is discussed in the remainder of this work.

II. Background

The visual code marker that is processed by the proposed algorithm has been devised by Rohs and is described in [3]. The visual code marker is comprised of three salient parts, which are the fixed guide bars, the fixed corner elements, and the data area, as illustrated in Figure 1. The purposes of the fixed elements as envisioned in [3] are also indicated on the figure.

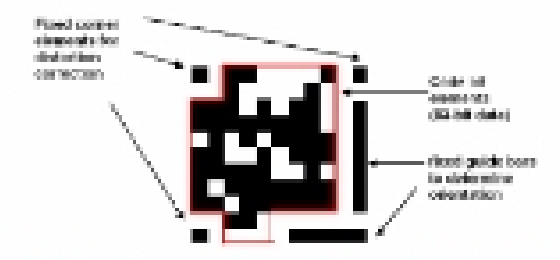

Fig 1. A typical visual code marker and salient features.

The vertical and horizontal guide bars have been intended for localizing and inferring the orientation of the code. The three squares at the corners are meant for detecting any distortion in the image. The data area encompasses the visual code pertaining to the actual code bits.

The visual code marker used in this study is an array of 11x11 elements, where an element is defined as a black or white square of the size of a fixed corner element. The vertical guide bar is 7 elements long and the horizontal guide bar is 5 elements long. The contiguous elements of the corner elements and the guide bar elements are also fixed and white.

III. Method

A. Algorithm Overview

The algorithm takes as an input a 24-bit colored image and performs the following operations on the image:

Marker extraction: The portions of the image that contain the visual code markers are detected and are isolated from the rest of the image.

Marker characterization: The parameters that characterize the stance of the visual code marker within the original image are extracted at this stage. These parameters are the rotation angle of the code marker and the position of its four corners.

Marker analysis: Given the rotation angle and the location of the four corners, the location of the code marker in the original image can be found and the binary data within can be extracted.

A system-level representation of the algorithm is presented in figure 2.

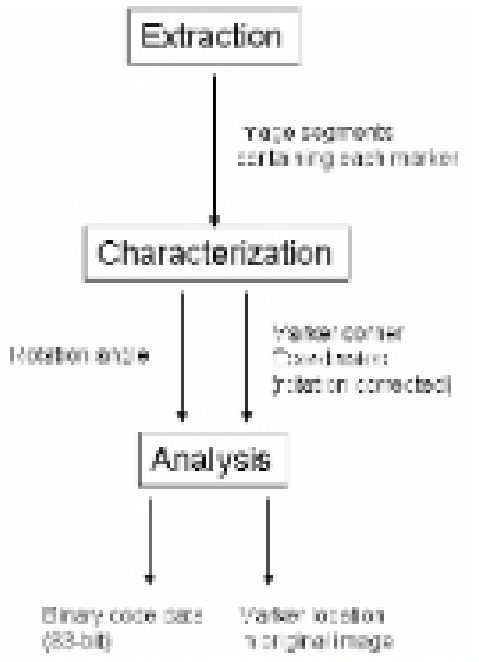

Fig. 2. Overview of the algorithm

We probe further into the mechanics of each operation separately in the following sections.

B. Marker Extraction

A flowchart illustrating the operation of the marker extraction algorithm is given in Figure 3.

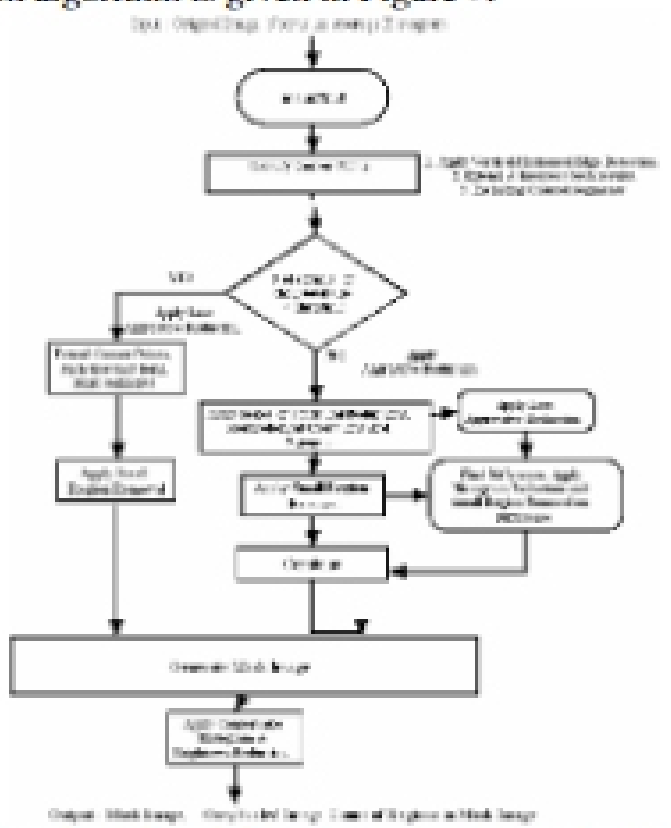

Fig.3. Flowchart describing the marker extraction algorithm.

As a first line of processing, the color images are grayscaled according to the scale-by-max algorithm, which scales each channel by its maximum component, as demonstrated in Figure 4. The generic idea in identifying the regions containing the visual code markers is corner detection. The primary motivation in applying this technique is that the visual code marker contains small white and black squares and is expected to contain a prominent number of corner points.

a)

b)

c)

Fig. 4 a) Original Image, b) Image after Scale By Max Applied and c)Image after gray scaling

To find the corners within the image, horizontal and vertical edge detection are applied separately and the intersection points of the resulting edges are found. Because of the non idealities at this stage extra processing is required to get 1 pixel thick corner points. Figure 5 illustrates a dilated version of these corner points.

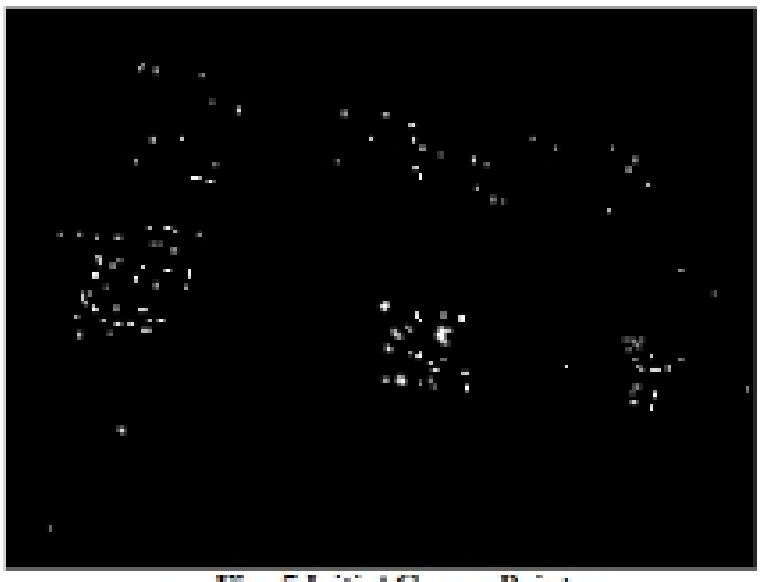

Fig. 5 Initial Corner Points

To attain 1-pixel thick corner points, the thickness of the detected lines are increased by dilation and then the intersection points are logically AND'ed.

It was observed that color could be utilized for a better estimate of corners; since the visual code markers are in black and white and any corners identified on the colored portions of the image can be discarded. However, the inherent chromatic noise makes color identification difficult. Algorithm 1 was utilized for the detection of colored regions:

Algorithm 1:

Grayscale the image by averaging 3 channels to get image 1

Grayscale the image by picking maximum of 3 channels to get image 2

For any pixel within image 1 and image 2
  If (image 1) – (image 2) > threshold then
    identify the pixel as a color pixel.
  Else
    identify the pixel as a black-and-white pixel.

If a pixel has a much higher channel value for any of its channels with respect to the averages of these channels, then one of the color channels dominates the other 2 channels, and the pixel is a colored pixel. A threshold of 40 was selected and was observed to be a satisfactory intensity separation. The colored regions identified on the example image of Figure 4 are shown in Figure 6. The corner points which corresponded to color pixels are logically subtracted from the image. (See figure 7.)

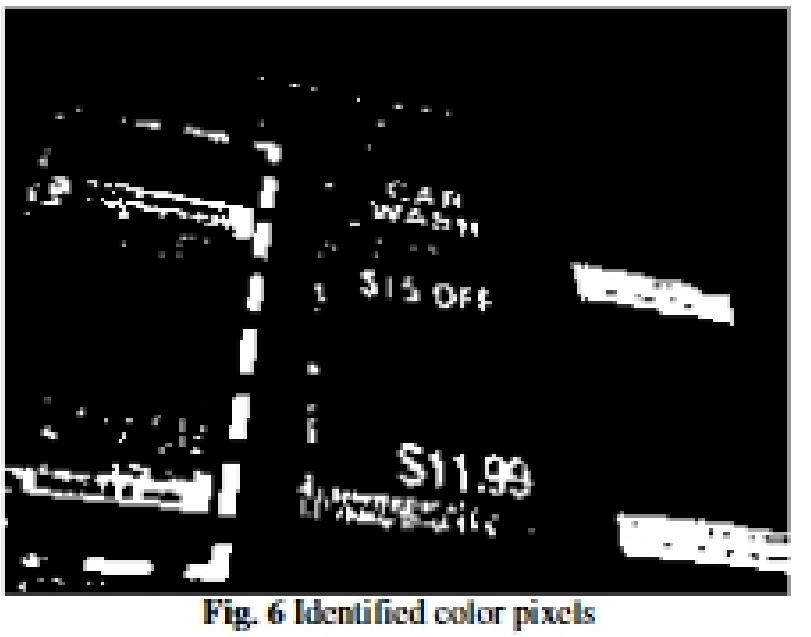

Fig. 6 Identified color pixels

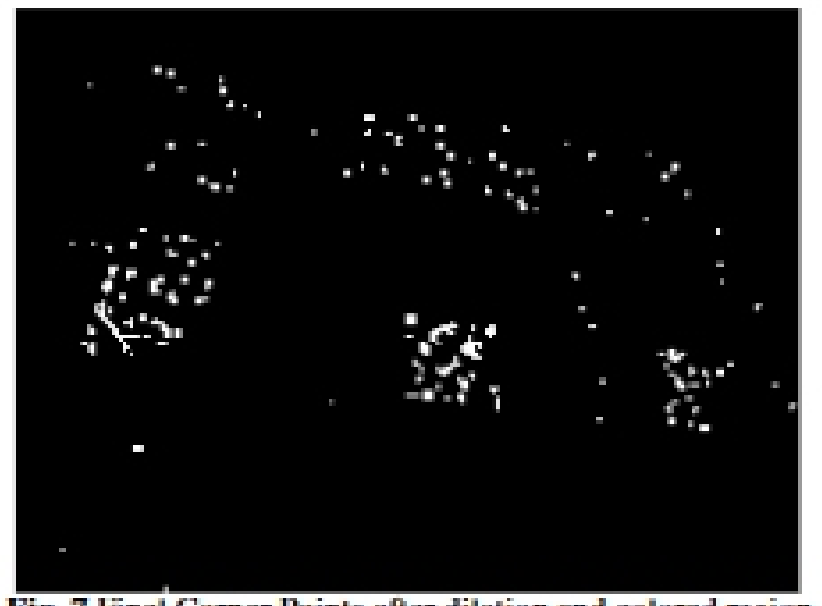

Fig. 7 Final Corner Points after dilation and colored region subtraction.

The corner points are next expanded until they combine and form larger regions. Once this is achieved, one of the two different strategies is chosen to identify which of these large regions the code markers are, and to expand these regions: For images that contain a small number of corner points or that have small regions after dilation a less aggressive algorithm is chosen. Otherwise, a more aggressive strategy is pursued.

The less aggressive strategy applies small region removal to eliminate the regions that contain a small number of corners and thus do not qualify as code markers. A final dilation can now be applied in confidence to combine the regions containing portions of a single visual code marker and to obtain a mask image. The output of the less aggressive strategy for the example image is illustrated in Figure 8.