Detection and Reading of Visual Code Markers

Kim M. Shultz

Abstract—**This paper describes a method for detecting and decoding visual code markers. Markers are located using erosion by a template to match guide bars. Once markers are located and verified, they are corrected for rotation and skew. Finally, the code is read by comparing the marker to a threshold intensity.**

I. Introduction

THE images are first put on a 0 to 1 scale and then translated into a gray scale. All processing is performed on the gray images. To identify the code markers, erosion is performed with a template to detect the guide bars on a dilated edge image and a thresholded image. Detected markers are verified by matching white lines on either side of the markers. Once markers are detected and confirmed, they are oriented so that the vertical guide bar is truly vertical. The corners of the marker are then identified. The contents of the marker are read by comparing blocks of the marker to a threshold.

II. Marker Identification

A. *Marker Location*

Code markers are identified by erosion using a template to match the code marker guide bars (see Fig. 1). The template is based on the guide bars for a given marker pixel size (the width or length of a single bit in the code). It consists of two lines, each one pixel thick. The horizontal line is 5 marker pixel sizes wide, to match the horizontal guide bar. The vertical line is 6 marker pixels long, to match the vertical guide bar. The lines are separated vertically by twice the marker pixel size. The vertical spacing is larger than present in the real guide bars, and the vertical bar shorter, so that the template can also identify a larger marker.

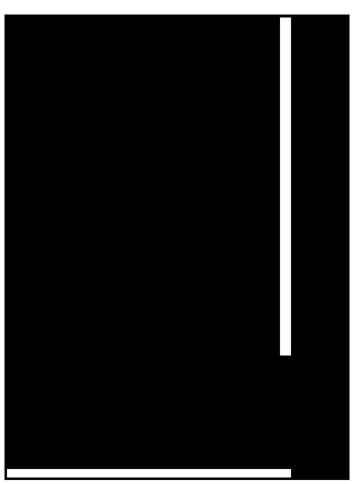

Fig. 1. Template used for erosion to find code markers

Templates of various sizes are used, since the size of the marker can vary. A marker of a given size can be matched by a template based on that marker size (see left of Fig. 2) or a smaller template, based on a marker of half the size (see right of Fig. 2). Using template sizes that vary by factors of two should therefore catch all of the markers.

The template is rotated to catch different orientations of the marker. The rotation is performed by rotating the marker by a growing angle. The angle increment is 8 degrees, chosen since the vertical line tipped at 8 degrees will still fit in the vertical guide bar (see Fig. 3). This rotated template will now extend beyond the horizontal bar, but a smaller template will not have the same problem. Therefore, the template sizes are not varied by a factor of two, as is needed for the size constraints, but instead by a factor of about 2/3, so the overlap will allow all marker sizes and orientations to be found.

The erosion is performed on a thresholded image and an edge image. The threshold image will ensure that the areas matched are dark (corresponding to the dark marker bars). The edge image will ensure that the areas matched are in fact near edges and not is large dark areas of the image.

Fig. 2. Largest (left) and smallest (right) template that will match a given marker size

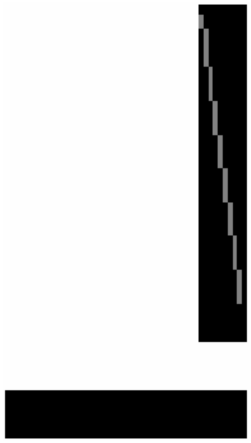

Fig. 3. Vertical template bar rotated 8 degrees matching vertical guide bar.

The edge image is created using edge detection with a Sobel filter. The edges are then dilated using a disk large enough to fill in the guide bars (based on the largest marker that will be located with that template size).

Before the thresholding is performed, the image must be corrected for lighting differences. Since the lighting may vary across the image, the lighting bias is calculated using a low pass filter. The size of the low pass filter is determined based on the size of the template. The size of the filter is twice the size of the largest code marker that can be detected with the template. The filtering is performed in the frequency domain (using an FFT on the image and the filter) for computational efficiency. The image is then thresholded by comparing the image to the low pass filtered image, with pixels in the image that are larger than the corresponding low pass filtered image set to 1.

The dilated edge image and the inverse of the threshold image are compared, saving pixels where they are both 1 (areas that are near and edge and dark). Erosion with the template is then performed on this combination image. Only areas that match the template will be left after erosion.

B. Marker Verification

When you submit your final version, after your paper has been accepted, print it in two-column format, including figures and tables. Send three prints of the paper; two will go to IEEE and one will be retained by the Editor-in-Chief or conference publications chair.

The results of erosion are used to determine candidate regions for further analysis. Since many corners in the image will return a positive result, the matches are tested for the presence of white space both inside and outside the suspected marker bars.

Starting with the horizontal guide bar, a line just above the matched bar is checked. The line is also shifted to the right from the matched horizontal bar, so that the check is sure to include the region between the vertical and horizontal bars. If all the pixels along the line in the thresholded image are white, the marker passes this check. If not, the line is moved up by one pixel and checked again. The line continues to move up until it finds a white line or it has moved more than three times the marker pixel size used for the template.

If a white line match is found above the horizontal guide bar, the test is repeated for the vertical bar, moving left instead of up. If that passes as well, the horizontal and vertical tests are repeated for the outside of the marker (moving down or right respectively) with a limit on the line movement of four times the marker pixel size used for the template.

If any test does not pass, the suspected marker location is rejected. If all four tests pass, the suspected marker location is recorded as a hit.

In order to avoid repeats, any suspected marker location within three times the template marker pixel size of a confirmed marker is rejected if both were detected at similar angles (within 16 degrees, two degree increments).

C. Search for Further Markers

Once a marker is found and analyzed as described below, the marker search process is continued until either three markers are found or 50 seconds have elapsed since the beginning of the analysis. This ensures that even with processing time after a marker is found, the program will finish within one minute.

III. Marker Processing

A subimage around each confirmed marker is saved for further processing, along with the location of the hit and the angle of the matched template. The lighting bias is removed. The subimage is rotated by the negative of the angle of the template that found that location to obtain a roughly correctly oriented marker.

A. Rotation Detection

To determine to exact angle of the marker, a subimage of the subimage is analyzed. This subimage is selected so that it will contain the entire vertical bar. Edges in this subimage are detected by subtracting the subimage from a dilation of that subimage. A Hough transform is then applied to the edge image to determine the rotation angle. The rotation angle is the angle with the largest peak in the Hough transform. The subimage with the entire marker is then rotated by the negative of this angle to obtain a correctly oriented marker.

B. Size Detection

To accurately determine the size of marker pixels, an area containing the entire vertical bar is analyzed. A vertical line the same height as the height of the image is used as an erosion element on the inverse of the thresholded image of the subimage. The size of the vertical line is decreased until a match is found. If the vertical guide bar is matched, the eroded image will have content at the center of the guide bar. Nearby matches in the eroded image are combined to create a single point, roughly in the middle of the guide bar.

To ensure that the erosion found the guide bar and not another image artifact, a hollow rectangle just larger than the supposed matched guide bar is checked in a thresholded image with a lower threshold (to avoid noise and blur). If all the pixels in the rectangle are white, the match is saved. If not, the rectangle is expanded one pixel in each direction and rechecked. This is repeated until a white match is found or the rectangle has expanded by a full estimated pixel size in each direction.

If the erosion match is not confirmed by the white rectangle check, the process of shrinking the erosion element is continued until a match is found or the size of the erosion element shrinks below twice the pixel size used to create the original guide bar template. If no guide bar is matched, the size cannot be detected, and the marker is not a correct marker. This acts as a final marker verification check. If the marker does not pass this verification test, no further processing is performed on it.

C. Skew Detection

Even after the marker is rotated so that the vertical guide bar is truly vertical, the marker may be skewed due to the angle of the camera when the image is taken. Therefore, this skew must be estimated. Based on the location of the vertical bar and the marker pixel size, a thresholded subimage that will contain the horizontal bar is analyzed. The edges of this subimage are detected by subtracting the thresholded subimage from a dilation of itself. A Hough transform is applied to these edges, and the angle with the largest peak is recorded as the skew angle.

D. Marker Corner Location

The corners of the marker are found using the correctly rotated subimage.

1) Bottom Right Corner: The horizontal location of the upper left corner is found by averaging the horizontal locations of all points that are contiguous with the vertical guide bar. These points are determined by using region labeling on the inverse of a thresholded image with a low threshold. The low threshold is used to prevent blurring from connecting the guide bar with other marker pixels.

The vertical location of the bottom right corner is estimated by moving five marker pixel heights down from the center of the vertical guide bar. If this point is dark, it is assumed to be part of the horizontal guide bar. The top and bottom of the horizontal guide bar are found by moving up and down until white is found in the thresholded image.

If the point five marker pixels below the vertical guide bar is white, the process begins at the center of the vertical guide bar. Moving down from the vertical guide bar center until first white then black is found gives the location of the top of the horizontal guide bar. Moving further down until white is found again gives the location of the bottom of the horizontal guide bar.

The average of the top and bottom of the horizontal guide bar is saved as the vertical location of the bottom right corner.

2) Top Right Corner: The top left corner is found in a similar manner as the bottom right corner. Part of the fixed corner pixel is identified either by moving five marker pixel heights up from the vertical bar center or, if that point is white, by moving up from the vertical bar center until white then black is found in the thresholded image with a low threshold. All points contiguous with that found black point are found by using region labeling on the inverse of the low-thresholded image. The average vertical and horizontal locations of these contiguous points are saved as the location of the top right corner. The vertical pixel size is re-estimated from the vertical locations of the right pixels.

3) Bottom Left Corner: To find the bottom left corner, the horizontal pixel size must first be accurately estimated. To do this, the width of the vertical guide bar is calculated by moving left and right from the vertical guide bar center. This width is the horizontal pixel size with blurring. The left end of the horizontal guide bar is calculated by moving left from the bottom right corner at the skew angle until white is found. The horizontal distance between the bottom right corner and the left edge is four and a half pixel widths plus half the blur. Combining this information with the width information from the vertical guide bar gives an accurate estimate of the horizontal pixel width.

The right edge of the bottom left corner fixed pixel is estimated by moving left from the bottom right corner nine and a half marker pixel widths and up or down an appropriate distance based on the skew angle. If this point is white in the low-thresholded image, the right edge of the bottom left corner fixed pixel is found by moving left at the skew angle. If the point is black in the low-thresholded image, the starting point is at that point. All points contiguous with that point are found using region labeling on the low-thresholded image. The locations of these points are averaged to find the bottom left corner location. The horizontal pixel size is re-estimated using the bottom corner locations.

4) Top Left Corner: The right edge of the bottom left corner fixed pixel is estimated by moving left from the bottom right corner nine and a half marker pixel widths and up or down an appropriate distance based on the skew angle. If this point is white in the low-thresholded image, the right edge of the bottom left corner fixed pixel is found by moving left at the skew angle. If the point is black in the low-thresholded image, the starting point is at that point. All points contiguous with that point are found using region labeling on the low-thresholded image. The locations of these points are averaged to find the bottom left corner location. The horizontal pixel size is re-estimated using the bottom corner locations.

From the starting point, a test location moves down and to the right. At each step, a vertical line between the test location and the white horizontal line is checked for black. If no black is found, a horizontal line between the test location and the white vertical line is then checked for black. If no black is found, the test location moves down and right again until black is found. Once black is found, all points contiguous with that point are found using region labeling on the low-thresholded image. The locations of these points are averaged to find the top left corner location.

This is the location of the origin in the subimage. The subimage is then un-rotated to its original state with the origin marked. The location of the origin in the overall image is extrapolated from the origin in the un-rotated subimage and the location of the subimage in the overall image.

E. Code Reading

The exact location of the marker in the subimage is determined from the corner locations. An image containing only the marker is created using a projection transform based on the corner locations.

The location of each data point is calculated by generating a code marker with only that data point and removing the fixed guide elements. The image with only the marker is then divided into 121 blocks. Each data point is matched with its corresponding block. The blocks are weighted bilinearly such that the center is the most important and the edges are de-emphasized, keeping the mean the same. A threshold is computed by finding the average each of the fixed black corners and an adjacent white pixel. Each weighted block is compared to this threshold to determine if it is white or black.