EE368 group21 report

Visual Code Marker Detection

EE368 Class Project, Spring 2005-2006

Jianrong Chen
Analog Devices Inc.
Wilmington, MA 01887
pierce.chen@analog.com
chenjr@stanford.edu

***Abstract*— With the improvement of technology, most mobile phones equip camera higher than VGA (640x480) resolution. Visual codes is one type of 2D graphic code used for the interaction with the mobile phone, to enable the interaction between real world and multiple interesting services and applications. Michael Rohs describes the use of visual code markers in his project. This paper shows the flow and algorithm used to detect the visual code markers.**

Keywords*— *visual code, detection.

Topic area—recongnition.

I. Introduction

The visual code markers is an 11x11 array, each element is either black or white. There are three fixed corner elements and two fixed guide bars. The larger and smaller guide bar are for determining the location and orientation of the code, three cornerstones are for detecting the distortion, and the data area with the actual code bits. There are total 83 bits of data, the code bits are protected by a (83, 76, 3) linear code with Hamming distance three, which encodes 76bits of data on an 83-bit codeword.

The visual code marker detection is to take an input image captured with cell-phone camera, doing detection in Matlab to identify the number of visual code marker (1~3), location (x-y coordinates of the upper left corner of each code marker, and also outputs the bits of each code marker.

II. Recognition Algorithm

The recognition algorithm performs the following main steps on the camera image described on [1], and output the code bits.

1. Convert color image into binary image with adaptive thresholding.
2. Region identification and cluster labeling on the binary image.
3. Characterize each cluster, get the information such as center location (x, y), range (max/min y, max/min x), ratio or shape of the cluster, and main angle of the cluster pattern.
4. For each cluster, search for the major guide bar by looking at its ratio and size. When it's confirmed, search for smaller guide bar and upper left cornerstone.
5. Information of cluster, ratio, angle, size, distance from two boundary pixels are used to determine whether the guide bar are found or not.
6. When two guide bars and upper right corner are found, search for the bottom left corner and upper left corner.
7. Based on the four corner coordinates, report the 83 data bits.
8. Search for next visual code marker.

A. Convert color image into binary image with adaptive thresholding.

To reduce the processing effort, the input camera image need be converted into BW image for the remained pattern recognition algorithm. First, a gray scaled image is generated from input color image. The Matlab function rgb2gray can be used here, to make it simple, the formula used by Michael Rohs in [1] is used here: *gray = (green + red)/2*. The blue information is ignored here because it does not contribute much in luminance component. Chrominance component is ignored at this step. Further test shows that without chroma information, some visual code marker is hard to be detected due to some useful information is dropped. Using the chroma value can help to remove some pattern before mathematics calculation, such as the human eye recognition process, colorful part is not the black/white marker we are looking for.

The second part is to convert the gray image into binary image for pattern recognition. Because that the brightness of the camera image is not constant and the visual code may not be illuminated evenly, an adaptive thresholding method [2] is used to find the threshold based on line scan from left to right and right to left alternative with an average window, also averaging current average value with the average value in last horizontal line at the same column location to reduce the effect of zigzag traversal fact every other line.

This thresholding method turns out very effective, it can handle most cases, but it's not perfect for some special case, which will be explained later.

B. *Region identification and labelling.*

From last step, the pattern to be identified becomes value 1 in the binary image, in this step all the connected pixels are labeled as regions with an ID, the labeled image is p(y,x). First phase the image is traversed row by row, assigned with a preliminary label, second phase is to merge the equivalence regions who have different label with single label.

The traverse processing can be combined with the thresholding step to save extra loop time. With Matlab, this step can be done with the function of "bwlabel".

```
% cluster/region labeling
[t2, num_cluster] = bwlabel(t1);
```

C. *Region Characterization*

For each region, the following cluster information is prepared for future operation:

(1) center X

(2) center Y

(3) maximum X

(4) maximum Y

(5) minimum X

(6) minimum Y

(7) ratio

(8) angle

(9) size

(10) label

For each region id I, find all the pixels in this region with Matlab command [r, c] = find(p==I); [r, c] contains all the coordinates of pixels, then size, center X/Y, max/min of X/Y are easy to get.

Information of ratio and angle are only useful for guide bar recognition, the following conditions are used to remove the background pattern. (1) the size of guide bar must be larger than a minimum value (size > 20) and smaller than a value (max_x – min_x < 200 and max_y – min_y < 200), (2) when boundary pixel (r = 1 or 480, c = 1 or 640) is in this region, it can not be the guide bar.

Angle of the major axis of cluster is calculated with Hough algorigthm, with the select [r c] set, a smaller 2D image is created to do Hough transform, the maximum angle is stored for this cluster.

Ratio is calculated from the equation of ellipsis described in [1].

D. *Cluster Recognition*

The flow of cluster recognition is:

(1) Find major guide bar candidate.

(2) Find second guide bar, and upper right cornerstone on the direction of major bar extension.

(3) Find bottom left cornerstone on the direction of second bar.

(4) Find the approximate location of upper left cornerstone, search around to get its accurate location.

To search for major guide bar, the information of each cluster collected in last step is checked one by one. Based on the shape of major guide bar (7:1 in ideal case), the ratio should be less than 0.2 from experiments. For the second guide bar, the ratio is less than 0.3.

If cluster(i) meets this requirement, then find the two boundary pixels on its major axis from the angle information, P1 and P2. Calculate the distance between them as Ry, then search for the next cluster on this axis direction under the maximum distance max_R = Ry * 11/7. Two clusters should be found, the one with larger size is identified as second guide bar, the other with small size should be the upper right cornerstone.

Some extra checks are done to make sure the real clusters are found:

(1) The angle difference between second guide bar and major guide bar should be around 90 degrees. (I use 45 degrees to allow some shape distortion).

(2) The second guide bar's size should 5/7 of major bar. (50% difference is allowed here).

(3) The cornerstone's size should be around 1/7 of major guide bar, no larger than 2/7.

Now the direction of major guide bar is defined from the second guide bar point to the cornerstone. Then the reference direction of second guide bar is that rotated by 90 degrees anticlockwise.

Similar as before, two boundary pixels of second guide bar are found to get the length Rx, and calculate the distance from center of second guide bar to bottom left corner is around Rx* 8/5, then find it along that direction.

Now three corner clusters are known, the approximate location P1(x,y) of upper right corner is calculated from bottom-left and the Ry * 10/7. Due to there might be some shape distortion, the P1(x,y) doesn't belong to a region, then a search algorithm is performed to search around for a region, the search algorithm starts from P1(x, y), do search for right, down, down, left, left, left, up, up, up, up, … until a region is found in an allowable range.

E. *Reports the data bits.*

From step D, four corner cluster's center can be map to the code coordinates as

(1) Upper left: (0, 0)

(2) Upper right: (0, 10)

(3) Bottom left:(10, 0)

(4) Bottom right: (10, 8), it's the center of second guide bar.

From these four points, the (x,y) coordinate of each data bits (j, i) can be calculated by bilinear interpolation. Then look at the majority region label at a 3x3 window centered at (x,y), if the majority label is 0, the output data bit as 0, otherwise output 1.

III. Implementation Notes

There are several points in the whole recognition affect the final evaluation performance.

1)Image shape distortion.

This affects the corner searching, the calculated location may be on the background. A smart search algorithm is very helpful to locate the correct cornerstone.

2)Adaptive thresholding

Uneven illuminance make the image thresholding more difficult. Although the adaptive thresholding algorithm is working well mostly, there is special case that it will convert the cornerstone pixel as background. This is due to the fixed averaging window size, and some advanced algorithm should look at the nearest pixels (such as 8 pixels), to identify the white board of the marker, then choose better threshold value for the marker codes.

3)Constraints that filter out unwanted pattern.

Using ratio to identify the guide bar candidate is very effective to save search time. And it's also useful to verify the cluster is guide bar or cornerstone.

Angle is helpful to recognize the guide bar cluster, especially to confirm the relationship between major guide bar and second guide bar.

IV. Conclusion

The whole recognition algorithm is tested with the 12 test images, and it gets total score as 1724. There are markers in one test image missed, which is because that after thresholding, the cornerstone is missed. Also few data bits are recognized wrong due to the same reason, the hole is closed. An advanced adaptive thresholding algorithm with adaptive window size is to be developed to handle this case.

This is a pretty interesting project that shows how the basic image processing algorithm and mathematic are linked to some real applications. Mobile phone with camera will make future life more interactive. The phone can capture some 2D marker can the server can provide more information related to the marker. It's so powerful and feasible to become truth in the near future.

References

[1] Michael Rohs, **Real-World Interaction with Camera-Phones** 2nd International Symposium on Ubiquitous Computing Systems (UCS 2004), Tokyo, Japan, November 2004

[2] Pierre D. Wellner, "Adaptive Thresholding for the DigitalDesk"

[3] R.C.Gonzalez, R.E.Woods, S.L.Eddins, "Digital Image Processing using MATLAB".

[4] EE368 course handouts.