ECE661 Computer Vision - Fall 2014 - HW7

Fu-Chen Chen chen1623@purdue.edu

1. Image Segmentation

In this homework since the intensity differences between the letters and the background are very obvious, so I convert color images to gray-level images and apply Otsu's Method on gray-level images to extract the foreground letters. After that I also apply Morphological operation "Opening" to delete some small false segmentation of background, then perform "Closing" to smooth the contour of foreground and compensate the holes in foreground.

2. Component Labeling

For component labeling, I apply one-pass raster scan to label all the components by the following rules:

For each pixel I scan,

$$\begin{cases} \text{If no neighbor has label, assign a new label.} \\ \text{If one of neighbor has label or all neighbors' labels are the same, assign this label.} \\ \text{If neighbor's labels are different, assign one of the labels and merge the other labels to this one.} \end{cases}$$

By doing this when two components intersect at the pixel I scan, if they have different labels I will merge them. Thus it ensures all the pixels in the same connected component have the same label.

I define the foreground as the 4-connected object, so the neighbor mentioned above includes up-neighbor and left-neighbor of the pixels I scan. While doing the labeling, I also record the bounding box and the centroid of components for next step.

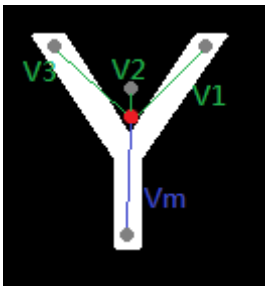
3. Shape Vector Extraction and Character Recognition

I extract the shape vector of letters in binary images by two methods: Harris Shape Vector and LBP Shape Vector. For each component(letter), I create a new binary image with 3-times width and height of bounding box of component and put the binary values of component to the center of this image. The new binary image allows me to extract the shape vector of one letter without the influence of other neighboring letters.

(a) Harris Shape Vector

For all the components(letters) in binary images, I apply Harris corner detector to detect all the corner points of letters. Each corner point and the centroid can form a 2D vector, and I define the main vector that has the max. length. The shape vector is defined by the intersect angles between vectors and the main vector. Since the numbers of corners in letters are different, the dimensions of shape vectors of letters are also different. By introducing the main vector and extract the shape vector with respect to the main vector, the shape vector is rotation invariant. It is also scale invariant since we only consider angle as feature.

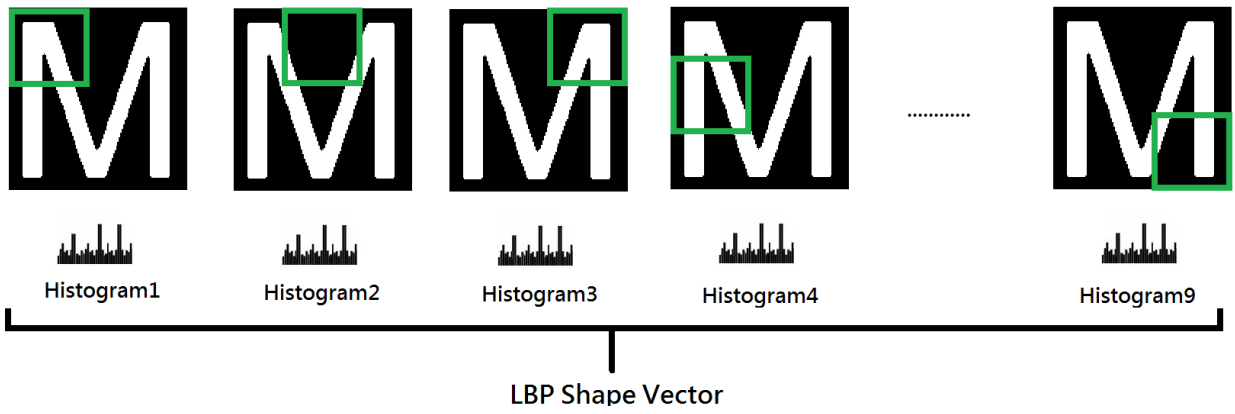
For example, the gray dots are the corners and the red dot is the centroid. The blue line is the main vector Vm(with max. length) and the green lines are the other vectors V1, V2 and V3. The shape vector S=[angle(Vm,V1), angle(Vm,V2), angle(Vm,V2)].



For character recognition with Harris shape vector, I compare the matching errors of all the training letters and the input letter and pick the letter with the smallest error. Since the dimensions of shape vectors are different (even they are the same letter), for each dimension of input shape vector I found its best match dimension in training shape vector, record the error and delete this dimension. If all the dimensions in training shape vector are deleted while there are non-matched dimensions in input shape vector, for each remaining dimension the error costs π(180 degree).

(b) LBP Shape Vector

Another way to describe the feature of letter is LBP(local binary pattern) histogram. For each component I divide its bounding box into 9 overlap regions, and generate the histogram of LBP code of each region. These histograms form the shape vector that each dimension in the vector is one of the bin in the histograms.



The LBP code could be defined as:

$$\mathrm{LBP(X)} = \sum_{i=0}^{N-1} s(X_{ni}, X) * 2^{i}$$

$$\mathrm{N\ is\ number\ of\ neighbor\ of\ X} = 8.$$

$$X_{ni}\ \mathrm{is\ the\ ith\ neighbor\ of\ X}$$

$$s(X_{ni}, X) = \begin{cases} 1, \mathrm{if\ pixel\ value\ of}\ X_{ni} > \textit{pixel value of } X \\ 0, \mathrm{otherwise} \end{cases}$$

pixel value

90	70	6
30	50	5
220	192	100

$\Rightarrow$

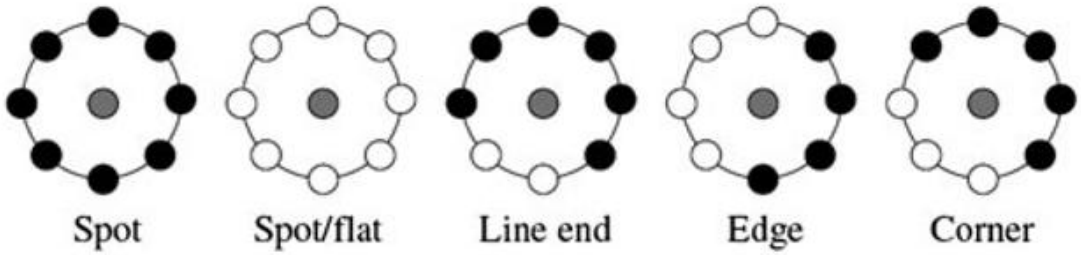
$s(X_{ni}, X)$

1	1	0
0		0
1	1	1

$\Rightarrow$ LBP(X) =11001110 (binary) = 206 (decimal)

which is a 8-bit value between 0~255.

Different LBP codes could present different types of edges or corners around the pixels, thus the histogram of LBP codes means the occurrence frequency of each type of edge or corner in the region(called local pattern), which could be a good feature for character recognition. For example, the black dot means the pixel value of neighbor is greater, while the white dot means smaller or equal. The distributions of dots(corresponding to binary value of LBP code) are the different types of local patterns.

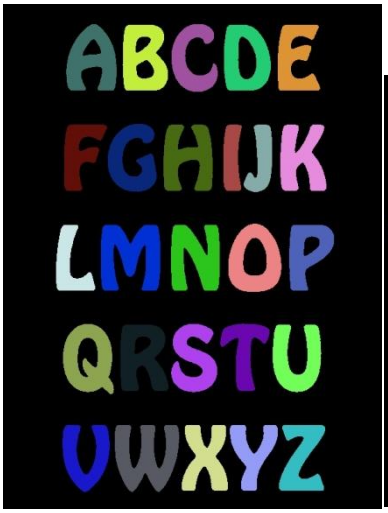


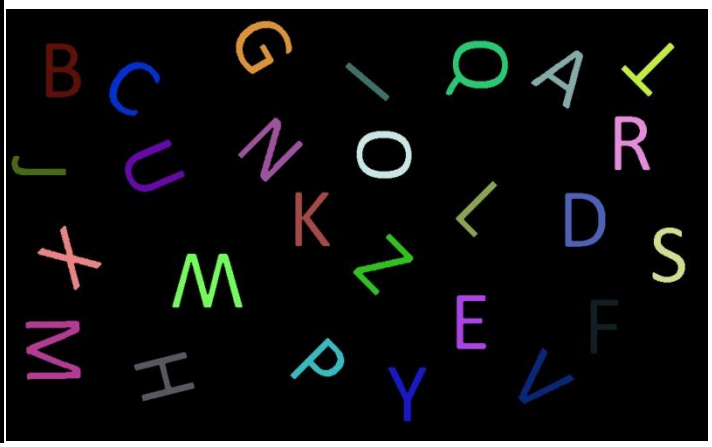
Since some of the local patterns are meaningless or caused by noise, I use uniform LBP code instead that is the subset of LBP code which only allows at most two 0-1 transition in binary LBP codes. For example, 00000000 & 11111111 & 00001111 & 11000111 full the requirement of uniform LBP codes, while 00100100 or 10000100 do not since they have more than two 0-1 transition. Beside the basic requirements of uniform LBP code, since I only want to extract the contour information of letters I also discard the code 00000000 and 11111111 that present the "Spot" or "Flat" of local pattern. By doing this the LBP histogram would not be effected by the thickness of letters. The final number of uniform LBP codes is 56, thus the dimension of shape vector is 9*56=504.

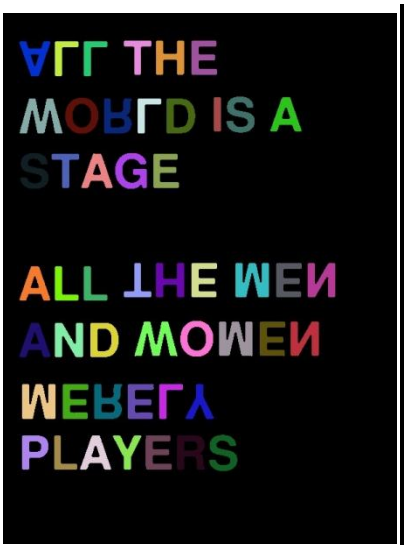
Because this kind of shape vector is not rotation invariant, for each training letter I rotate its binary image to 12 kinds of degree(0, 30, 60, 90.....) and generate its shape vectors. When doing character recognition I compare the L1 norm of input shape vector to all the training vectors of letters and degrees then pick the best one as the recognized letter. For scale invariant I normalized the histogram value by converting them to the percentage value, which presents the ratio of local patterns in the regions.

4. Results

(a) Labeled components







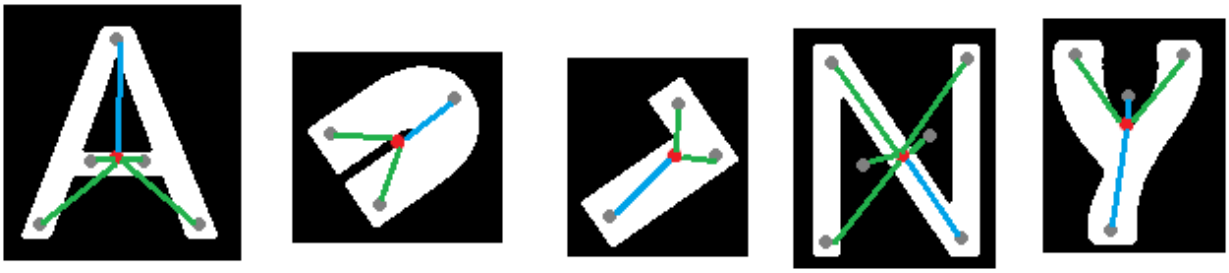
(b) Harris Shape Vector

Average recognition accuracy

	img1	img2	img3	img4	img5	img6	All
TRUE	7	2	3	9	5	1	27
FALSE	19	24	23	40	21	8	135
Accuracy	26.9%	7.7%	11.5%	18.4%	19.2%	11.1%	16.7%

(The accuracy of this shape vector is not good, so I don't present the accuracy per letter which is meaningless for comparing the accuracy between letters)

Corner points of letters



(C) LBP Shape Vector

Average recognition accuracy

	img1	img2	img3	img4	img5	img6	All
TRUE	17	10	24	44	24	4	123
FALSE	9	16	2	5	2	5	39
Accuracy	65.4%	38.5%	92.3%	89.8%	92.3%	44.4%	75.9%

Accuracy per letter

	A	B	C	D	E	F	G	H	I	J	K	L	M
T	9	3	3	5	11	3	4	5	2	2	2	10	6
F	1	1	1	2	1	1	1	2	3	2	2	3	1
A	90.0%	75.0%	75.0%	71.4%	91.7%	75.0%	80.0%	71.4%	40.0%	50.0%	50.0%	76.9%	85.7%

	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
T	6	9	4	3	3	7	5	3	1	6	4	5	4
F	1	0	1	1	4	0	2	1	3	1	0	2	0
A	85.7%	100.0%	80.0%	75.0%	42.9%	100.0%	71.4%	75.0%	25.0%	85.7%	100.0%	71.4%	100.0%

5. Observations

- Although Harris Shape Vector is rotation and scale invariant, it is inconsistent since we could not control the number of corners of each letter. The location of corner may also be different of the same letter in different images. Thus I think it is not a good feature for character recognition. Finding the main vector is also a problem since some letter are symmetrical, so we may find a main vector in one side while find another main vector in other side in different letter.
- The LBP Shape Vector provide more robust since it provides normalized statistical feature of characters regard their overall distribution of edges and corners. But one problem is that it is not rotation invariant so we have to compute all the shape vector of degrees for training. It could be better if we could find a method to define the main angle of characters then rotate each characters regarding this angle. Another problem is it extract the contour information of letters, but since the contour of letters in image 1, 2, 3 are not as straight as the training letters, the performance of LBP decrease in these images. It could be better if you find the skeleton or the center lines of letters, then apply LBP shape vector for recognition without the effect of contour.