An Approach to Chinese Number Recognition

Jason Rupard
Department of Computer and
Information Science
College of Computing, Engineering,
and Construction
University of North Florida
jrupard@gmail.com

ABSTRACT
This paper describes an approach to the symbol recognition of Chinese numerals. This approach uses techniques well established in digital image processing, including black/white conversion, digital binary morphology, pixel histograms for line counting, Zhang-Suen thinning for skeletonization, and other simple techniques such as median filtering. These techniques are combined to form a feature extraction algorithm. Once features are identified simple table lookups are preformed for number recognition. The table is constructed from known features in each of the non-complex and common Chinese numerals. Finally, results and conclusions are provided in regards to this algorithm along with ideas for future research.

Keywords
Image processing, symbol recognition, feature extraction, connectivity, thinning, pixel histograms, Chinese numbers

1. INTRODUCTION
Symbol recognition is a well-known discipline of digital image processing. From autonomous robots capturing images and locating clues on which actions to proceed with, to scanning programs processing documents and converting image text into actual ASCII text, symbol recognition techniques exist in many useful real world applications. This paper focuses on one particular type of symbol recognition and one correlating approach: the recognition of Chinese numbers [3,5]. This approach will attempt to recognize printed as well as hand written Chinese numerals. The recognition will be based on simple Chinese numbers in a common font that most newspapers use, rather than the complex numerals used on checks, banknotes, and coins. The numerals 0 thru 10 are shown in Figure 1.

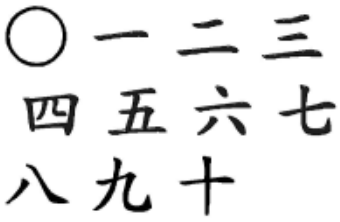

Figure 1. Simple Chinese numbers 0 – 10

2005 CCEC Symposium, April 13, 2005, Jacksonville, Florida, USA.

The algorithm used to recognize a numeral is based on what the author calls feature extraction. Feature extraction is composed of two feature types, point and line types. Feature point types are defined to be endpoints, corner points, 3-way intersections, and 4-way intersections within a numeral. Feature line types are defined as vertical and horizontal lines that make up the number. Once these features are identified, a simple table lookup is preformed. The table is constructed from the known features that exist in each Chinese numeral.

The feature extraction algorithm consists of three overall phases; pre-processing, feature extraction, and post-processing. The pre-processing phase is used to set the image for feature extraction through the use of two techniques: simple black/white conversion and binary morphology. The feature extraction phase uses a line scanning technique for discovering feature lines, Zhang-Suen thinning, and finally connectivity calculations for feature point discovery. The post-processing phase includes orientation, point reconciliations, and feature matching via table lookup. The rest of this paper will describe these techniques in detail with results and conclusions provided afterwards.

2. Feature Extraction Algorithm
For simplicity's sake, the feature extraction process in this paper will focus on one Chinese numeral in an image at a time. The image should come into system in gray-scale form. Each pixel in a gray-scale image is represented as a byte of data and therefore can be a gray value between 0 and 2^8-1. A black pixel will have value 0 and a white pixel will hold value 255. Every other value represents a gray shade between white and black [1]. The algorithm expects the numeral to be black, foreground color, and the background color to be white.

2.1 Pre-processing

2.1.1 Black/White Conversion
The first step of pre-processing is to take the gray-scale image and convert it to a black and white, binary image. A binary image pixel has only two values, 0 for black and 1 for white. To convert gray-scale to black and white a simple threshold technique is used; a pixel value greater-than the threshold value becomes white, a value less-than becomes black [1]. This technique is used to remove artifacts sometimes introduced by the scanning process. Also, a binary image is required for the remainder processes.

2.1.2 Binary Morphology

The second and last step in the pre-processing phase is a binary morphology called closing. This process is used for filling tiny holes within the numeral that can be introduced when the number is hand written. It is important that a solid numeral is used in later processes, otherwise they may fail. The closing operation is a combination of two binary morphologic operations called dilation and erosion, in that order.

Both the dilation and erosion operators take two inputs: the image and a 3x3 structuring element of pixels, shown in Figure 2. A 3x3 structuring element is a matrix with the center as an anchor point. Both operations use the structuring element by superimposing it over a pixel in the image where the anchor point of the 3x3 is imposed directly over the pixel in question. The dilation operation superimposes the structuring element over every background pixel, or pixel with value 1 (white). If at least one pixel in the structuring element coincides with a foreground pixel with value 0 (black) in the image underneath, then the current pixel is set to the foreground value. The erosion operation uses the same structuring element on each of the foreground pixels. If for every pixel in the structuring element, the corresponding pixel in the image underneath is a foreground pixel, then the current pixel is left alone, otherwise it is set as a background pixel [6,7]. It is important to understand that when the value of a pixel is changed in either operation, the change must be recorded and then carried out after the operation is completed, or the change must be altered on a temporary image which is later returned by the operation. Otherwise a rippling effect of incorrect values will occur from the operations.

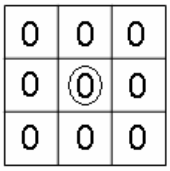

Figure 2. 3x3 Structuring Element

To close larger holes in the numeral, a sequence of N dilation followed by N erosions can be used. As N increases, the size of the hole that can be filled increases. The effects of closing are shown in Figure 3. The image is now ready for the feature extraction phase.

Figure 3. Closure operations before and after

2.2 Feature Extraction

2.2.1 Line Scanning

The line scanning process returns the number of vertical and horizontal lines that make up the numeral. These two numbers are recorded as the feature lines of the numeral and are used to determine the final output. Since the process of determining the number of vertical and horizontal lines is the same, except for switching the dimensions that are being scanned, we will focus on finding just the number of horizontal lines.

The first step is to construct a horizontal foreground pixel histogram [1], shown in Figure 4. In this case, the histogram plots the frequency of black pixels in a row of the image. Next, the histogram is smoothed out by keeping only the major peaks and valleys. This is accomplished by running a 5x5 median filter over the histograms image [1,8]. Also, closing is used to eliminate very small sub-valleys within larger valleys. Finally, the histogram is ready for line scanning.

Figure 4. Numeral seven with its horizontal pixel histogram

Starting off at the top of the histogram's image, scanning across the pixel frequencies counts the number of black/white or white/black changes. This scanning continues down the histogram recording the number of changes at each increment. As soon as the current number of changes on the scanned line is less than the previous line, the scanning stops. The number of horizontal lines in the numeral is equal to the number of changes in the previous scanned line divided by 2. Figure 4 shows the end location of the scanning line. In this figure, the "top" of the histogram is the right side and the "bottom" is the left side. Therefore, this scanning process would be from right to left. Note that the horizontal pixel frequencies could have been plotted along the x-axis of the histogram in Figure 4. Plotting along the y-axis was chosen for visual affect.

2.2.2 Zhang-Suen Thinning

Thinning is a process for producing the "skeleton" of an object in an image. The proceeding steps of the feature extraction algorithm work with the skeleton form of the Chinese numeral. Figure 5 shows the numeral five and its skeleton or thinned version. Many thinning algorithms exist. For this approach, Zhang-Suen was chosen because of its simplicity and speed [9].

Figure 5. Numeral five and its skeleton

Thinning is the process of stripping away outer pixels until no further pixels can be stripped without losing the overall shape of the object. Before explaining the Zhang-Suen algorithm we

must define a couple of parameters—a 3x3 neighborhood and its connectivity. The 3x3 neighborhood of an image pixel (I(i,j)) consists of its eight adjacent pixels. Figure 6 shows the 3x3 neighborhood of I(i,j), where I(i,j) is P_1. Connectivity of a neighborhood is defined as the number of object parts in that neighborhood. If connectivity is 1, then the removal of the center pixel in the neighborhood will not affect the overall structure of the object. If connectivity is 2, then the removal of the center pixel will spilt the object into 2 separate parts, and so on for 3, 4, etc. To find the connectivity of a 3x3 neighborhood, visit pixels P_2, P_3 ,..., P_8, P_9, P_2 counting the number of white-to-black changes. The number of changes is the connectivity.

P_9 $(i-1, j-1)$	P_2 $(i-1, j)$	P_3 $(i-1, j+1)$
P_8 $(i, j-1)$	P_1 (i, j)	P_4 $(i, j+1)$
P_7 $(i+1, j-1)$	P_6 $(i+1, j)$	P_5 $(i+1, j+1)$

Figure 6. 3x3 pixel neighborhood of I(i,j)

The Zhang-Suen algorithm iterates over the image pixels until no more pixels can be removed. Each iteration is broken up into two sub-iterations, as follows:

First sub-iteration:
Pixel I(i,j) is marked for deletion if ALL of the following 4 conditions are true

1. Connectivity = 1
2. Has at least 2 object neighbors and not more than 6
3. At least one of P_2, P_4, P_6 is a background pixel
4. At least one of P_4, P_6, P_8 is a background pixel

Delete marked after iteration completed

Second sub-iteration:
Same as the first except rules 3 and 4 have been changed

3. At least one of P_2, P_4, P_8 is a background pixel
4. At least one of P_2, P_6, P_8 is a background pixel

Delete marked after iteration completed

If at the end of any sub-iteration there are no pixels to be deleted, the skeleton is complete [9].

To finalize the skeleton, one last step is taken called staircase removal. The skeleton produced by the Zhang-Suen algorithm is one pixel thick in most areas, but not in all [7]. However, later in the process, the feature point detection algorithm requires a skeleton one pixel thick throughout. Holt's staircase removal fulfills this requirement by removing the extra pixels [4]. Figure 7 shows an example of a staircase condition. One of the two pixels in the middle row can be removed without affecting the connectivity of the object.

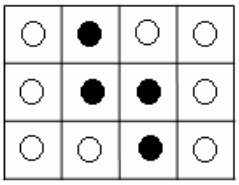

Figure 7. Staircase condition

To accomplish the removal, each foreground pixel's 3x3 neighborhood is compared to four structuring elements that are shown in Figure 8. X's represent "don't care" conditions, where the background/foreground value does not matter. If the current object pixel neighborhood matches one of the structuring elements and one of the X side neighbors is a background pixel, then the pixel stays. Otherwise, the pixel is marked for deletion. A side neighbor is one of the following, P_2, P_4, P_6, or P_8 referring to Figure 6. At the end of the iteration, pixels marked for deletion are set to background pixel values.

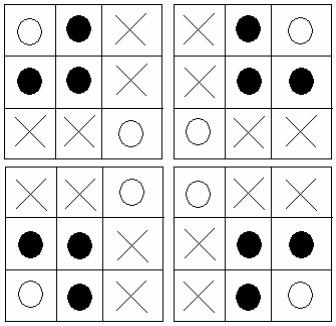

Figure 8. Four structuring elements for staircase removal

2.2.3 Feature Point Detection

For the final step in the feature extraction phase, feature points will be recorded for table lookup. The feature point detection function takes the skeleton of the numeral as input. Since the skeleton is only one pixel thick throughout we can use the connectivity function to identify endpoints, 3-way intersections and 4-way intersections. Corner points cannot be identified by connectivity. The process of identification tests each skeleton pixel. If the connectivity of the pixel is one, then an endpoint is recorded. If the connectivity is three, then a 3-way intersection is recorded. If the connectivity is four, then a 4-way intersection is recorded. When a pixel is recorded as one of these three types, the xy location of that pixel in the image is saved. This xy location is used later in the post-processing phase to reconcile points in very close in proximity of each other.

Notice, corner points were left out of the detection since a connectivity of two does not give any information on shape. A pixel with connectivity of 2 could be on a straight line, sharp corner, or even a gradual curve. Corner point detection should be the subject of further research.

2.3 Post-processing

2.3.1 Feature Scaling

Since a numeral can be hand written in different sizes, the 2D scaling transformation of all feature points to an arbitrary constant size is used to maintain uniformity [2]. To accomplish this, a bounding box is constructed around the Chinese number. The bounding box is defined as two xy points—the upper-left and lower-right corners. These points are assigned so that the box is, at most, encompassing the numeral. Next, the bounding

box is scaled, along with all the set points inside of it, to some constant size that all numeral boxes will be scaled to. Smaller numerals drawn are scaled up and larger numerals are scaled down to this constant sized box.

2.3.2 Feature Point Reconciliation

The last phase corrects minor errors introduced by some of the previous processes. These errors can range from artifacts introduced by the thinning of thick ends points (Figure 9a) or corners (Figure 9b) to the separating of 4-way intersection into two 3-way intersections (Figure 9c). Other errors, caused during handwriting of the numerals, are also corrected when this method tries to connect very close endpoints that were meant to be connected together (Figure 9d).

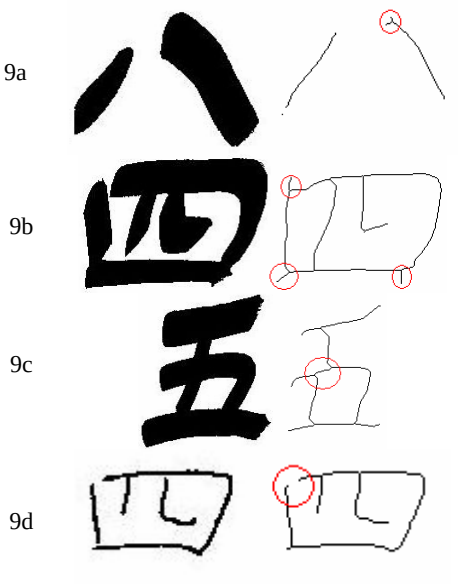

Figure 9. Possible errors introduced by the feature extraction algorithm

Each of these corrections uses a function that measures the distance between two or more xy feature points on the image. The problem suggested in Figure 9a is corrected by combining two or more endpoints that are a distance d_1 away from one 3-way intersection, into one single endpoint. The 3-way intersection is also removed. Figure 9b is corrected by removing an endpoint that is a distance d_2 away from one 3-way intersection. The 3-way intersection is removed also. A corner point would be recorded in place of these points. Correcting Figure 9c is accomplished by combining two 3-way intersections that are distance d_3 away from each other into one 4-way intersection. Figure 9d is corrected by deleting the two endpoints if the endpoints are a distance d_4 away from each other and a shortest path does not exists between them. Distances d_1, d_2, d_3, and d_4 are chosen by the programmer arbitrarily. To improve proper distance calculations, a method should be discovered that bases the distances on the scale and the thickness of the numeral.

2.3.3 Table Lookup

Table lookup is the final step in the Chinese numeral recognition algorithm. The table is constructed from simple Chinese numbers found in Figure 1. The feature points and lines that make up these numerals are entered into the table. For example, the entry of Chinese numeral 5 would have four endpoints, four 3-way intersections, zero 4-way intersections, three horizontal lines, and two vertical lines. Feature points are the primary lookup, while feature lines are secondary. In other words, if a perfect match is made on feature points, the matching is complete. Otherwise, if multiple matches are found for feature points, feature lines are compared. If the numeral is sill ambiguous after the secondary lookup, the set of possible matches is returned.

3. Results and Conclusions

Based on experimentation, this approach provides reliable, albeit not perfect matches. As one can see from the figures in Appendix A, size and location of the numeral in the image do not matter for either of the table lookups, but rotation does matter for the secondary lookup of feature lines. The detection algorithm does not account for rotated numerals within the image. Also, some numbers end up being ambiguous with just the feature point and line information recorded. Specifically seven, nine, and ten are similar if drawn correctly.

Six sets of hand drawn numerals were used for experimentation, 66 total numerals. The figures found in Appendix A are examples taken from the sets. Each set was drawn differently, but all were based on the numbers in Figure 1. The algorithm found an exact match for 54 of the 66 numbers. If more then one possible match is found the algorithm will return the possible set. Out of the twelve numerals not having an exact match, eight of them had correct possible sets. Only 4 of the 66 numerals were found to have an unknown or incorrect match. This algorithm's runtime is bounded by a polynomial function. The average runtime for the set of 66 numerals was 68.4ms with a standard deviation of 36.4ms. The test machine is a P4 2.4GHz with 1GB of memory.

Further research to improve this algorithm should focus first on corner point data which could provide perfect matching for all numerals. Other techniques to improve this approach might consider the location of a feature point relative to other points in the numeral. Overall this approach presents a solid building block for better Chinese number recognition in future research.

4. ACKNOWLEDGEMENTS

This approach was designed for the final project in Dr. Yap Siong Chua's graduate image processing course. A special thanks to him and Regeana Gibson for their support.

5. REFERENCES

[1] BOVIK, Al. 2000. *Handbook of Image and Video Processing.* Academic Press.

[2] HEARN, D., AND BAKER, M.P. 1997. *Computer Graphics, C version.* Prentice Hall, Upper Saddle River, NJ.

[3] HILDEBRANDT, THOMAS H., AND LIU, WENTAI. 1993. Optical recognition of handwritten Chinese characters: Advances since 1980. *Pattern Recognition.* Volume 26. 2:205-225.

[4] HOLT, C.M., STEWART, A., CLINT, M. AND PERROTT, R.H. 1987. An Improved Parallel Thinning Algorithm. *Communications of the ACM.* Vol. 30. 2:156-160.

[5] LEUNG, C.H., AND SZE, L. 1997. Feature selection in the recognition of handwritten Chinese characters. *Engineering Applications of Artificial Intelligence.* Vol. 10. 5:495-502.

[6] Morphology – Closing. 1994. *http://www.cee.hw.ac.uk/hipr/html/close.html*

[7] PARKER, J.R. 1996. *Algorithms for Image Processing and Computer Vision.* John Wiley & Sons, New York, NY.

[8] Spatial Filters – Median Filter. 1994. *http://www.cee.hw.ac.uk/hipr/html/median.html*

[9] ZHANG, T.Y., AND SUEN, C.Y. 1984. A Fast Parallel Algorithm for Thinning Digital Patterns. *Communications of the ACM.* Vol. 27. 3:236-239.

APPENDIX A

Various image numerals and their results:

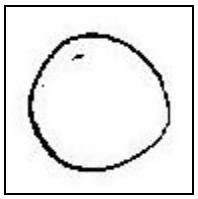

Figure 10. Result - primary match 0

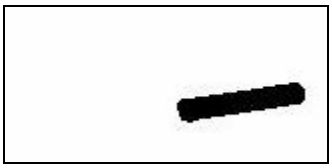

Figure 11. Result – primary match 1

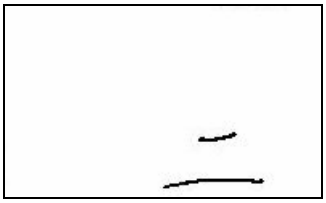

Figure 12. Result – secondary match 2

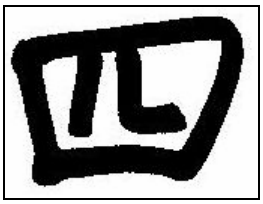

Figure 13. Result – primary match 4

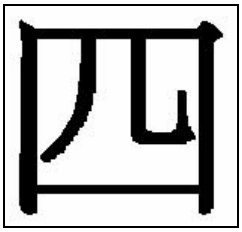

Figure 14. Result – no match

Figure 15. Result – primary match 5

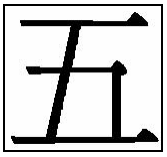

Figure 16. Result – primary match 5

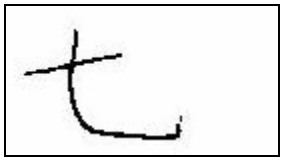

Figure 17. Result – secondary match 7

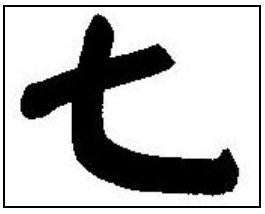

Figure 18. Result – possible 7, 9, or 10

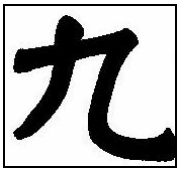

Figure 19. Result – secondary match 9

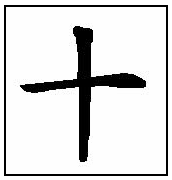

Figure 20. Result – possible 7, 9, or 10