Chinese character block recognition by DFT projection methods

Tommaso Torelli
SWIFT Division
KLA-Tencor Corp.
Milpitas, CA
tommaso.torelli@gmail.com

***Abstract*—We present an alternative method to address the challenge of positive identification of Chinese characters in dense text images. This method could be utilized as a front end interface to an Optical Character Recognition engine and in fact is applicable not only to Chinese, but to any language that utilizes ideograms or symbols whose distribution has a regular Cartesian arrangement.**

Keywords—Chinese; OCR; DFT

I. Introduction

The challenge of Optical Character Recognition (OCR) of Chinese text has been widely recognized to be one of very high complexity for a multitude of reasons [1][2]. Beyond the obvious fact that it contains as many as 20,000 distinct characters, with standardly used approaches it is also difficult to precisely isolate each character in an image filled with dense text, especially in the presence of natural clutter, such as white noise, that is commonly present in images. Attempts at removing or filtering the noise will often lead to modification or even removal of portions of the Chinese characters. The subsequent semantic interpretation operation, which involves matching against a known database of character images, can thus lead to mismatch or outright failure.

II. BASIS OF THE IDEA

A. Divide-and-Conquer Approach

The method presented here is based upon a simple divide-and-conquer idea. The complexity of Chinese text OCR can be greatly reduced by keeping the front-end problem of individual character identification as simple and general as possible and applying more sophisticated algorithms only at the individual character level, as opposed to the entire text image where: a) the possibility of false positives is much greater and b) any bias introduced by sophisticated algorithms would likely have more drastic effects.

Assuming that the front-end image processing can be kept quite simple, the added benefit of this approach is that it would certainly decrease the computational demand, since the more expensive operations are now deferred to only sub-portions of the image (i.e., the individual characters). Also the outcome of such an approach should be more robust to variations in the input image.

B. Abstract model of text layout

The method chosen in our approach starts from a very basic assumption of the model of text that this application is supposed to be used for. The model assumes that text symbols are laid out in a regularly-spaced, orthogonal grid. This effectively mimics the layout found most commonly in book, newspapers and other such widespread media sources. However, it even goes one step further: it allows the text grid to be broken down into sub-blocks, as long as they are all fairly aligned (within a few pixels) with respect to each other. A possible schematic depiction of such model could look as follows:

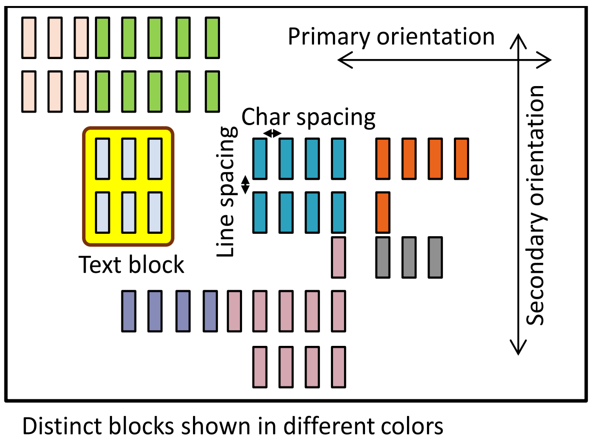


Fig. 1. Abstract model for Chinese text image assumed for this method.

III. METHODOLOGY

A. Discrete Fourier Transform (DFT)

With this premise, it is easy to see that this model lends itself well to adoption of a Fourier-transform based approach [5]. It has been used by others in similar applications [4], however more on the back-side of the OCR problem (i.e., for removal of noise when trying to match individual characters against a known database).

The DFT methodology is desirable for the following reasons:

- There are many available libraries for fast computation of discrete Fourier transforms (FFT). These are easily extended to 2 dimensional objects such as images since it is a separable filter.
- Multiple levels of information can be extracted from FFT data, such as amplitude, phase, periodicity, etc.
- Specifically for the purposes of this problem, we are interested in finding the primary peaks of the FFT at orthogonal orientations.
- The peak amplitude provides information about which of the two orientations is dominant, therefore it tells us whether the text is vertically or horizontally laid out, both of which are possibilities with Chinese text.
- The peak location in the DFT periodogram provides information about the spacing between successive character lines or individual characters within a line.
- Since we only care for very low-frequency repetition patterns (the text symbols or characters), this could be exploited to further speed up the calculation.
- Finally, by finding the orthogonal peak locations one can also at the same time obtain information about whether the angle of rotation of the input image (if off the 0- and 90-degree axis).

Having this information greatly simplifies the task of finding character lines and the individual characters within each line. Most of the processing that follows has linear computation complexity dependency upon the individual image dimensions, which is to say nearly trivial considering the computational resources that we can avail ourselves with nowadays.

B. *Image projection profile analysis*

The initial approximated DFT estimate on image angle, line and character spacing is a fully adequate starting point for taking image projections along the two major image orientations.

- The primary orientation represents lines of characters (in the natural order that they are typed or read by a user). Projections along this direction will have sharp peaks (or dips in case of white text on dark background) in correspondence of the handful of pixels spacing between successive lines. Identifying such peaks at a periodicity that closely matches the FFT predicted spacing is not difficult, although it requires one to come up with good criteria and some threshold tuning to make the approach robust to many different input image types.
- Once the character line locations have been finalized, the image can be now subdivided into individual slices, one for each character line. Within each slice, projections can now be taken along the secondary orientation, representing the individual characters themselves. Here again, peaks (or dips) will correspond to spaces between the characters. As long as the individual symbols are fairly equally spaced, a good match will be found when comparing to the FFT predicted spacing. This assumption does not hold for the Western alphabet, but is generally true for many Asian languages, including Chinese.

C. *Image pre-processing*

We deliberately tried to steer away from application of too many image pre-processing operations to reduce noise (such as median filtering and similar). There are two motivations behind this choice: A) the first is due to the possible loss of fidelity in the image, which increases the potential for failures down the processing chain; B) the second and actual motivation is that because of utilization of the DFT in the low-frequency ranges is much less impacted from conventional noise sources in the image (such as white noise).

The only pre-processing steps that were applied are as follows:

- Conversion from color to grayscale: most of the useful information in the image for this application is anyhow preserved. The color information could have unique value in addressing images with low-contrast text relative to the background, but this would require additional user feedback (such as pointing to the image area where text is located) and we deliberately tried to implement a solution that works as a black box, where the only input is one or multiple images (for batch processing).
- Image conversion to double, followed by normalization to max. Clearly this operation does not cause any further loss of information.
- Localized Otsu thresholding: this step is optional, but recommended in cases with less than ideal contrast or much clutter in the background. Generally speaking, application of this step helped improve the quality of the FFT, thus getting more accurate peak locations and ultimately a higher positive hit rate. The impact of this additional step to processing time was not too bad (roughly added 10% to the total time). Default values for tile size are 16x16, with a standard deviation cutoff value in the tile of $\sigma = 0.1$ to arbitrate between the choice local and global threshold values.

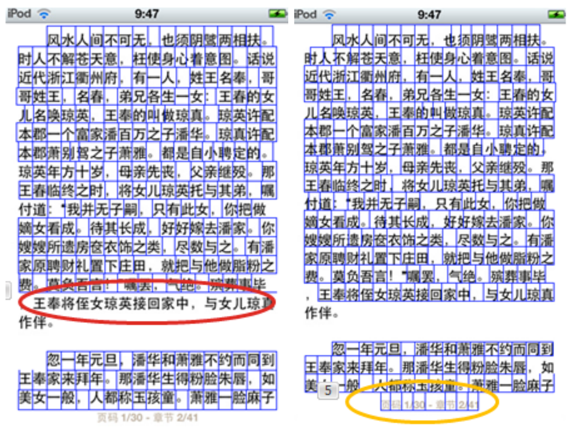
IV. EXPERIMENTAL FINDINGS

The proposed approach is found to work quite well for the type of text layout model that was described at the outset. We have gathered several text images by using the Google images site, by searching against terms such as “Chinese text”, “Ideographic text”, “Rotated text” and so on. We have then subdivided these images into five main categories:

- Regular Blocks: images where independent text blocks are nonetheless well aligned with respect to each other. In the simplest cases, it could be a page full of regularly arranged Chinese characters, but many are not.
- Irregular Blocks: here we have examples of image where independent blocks are not self-aligned. We can start seeing how for such cases this approach only works well in some parts of the image, but breaks down in others.
- Noisy Background: these are images that have both irregularities in the block layout and have not only text, but also pictures and random clutter in the background. Here, except for the simpler cases, this approach will not be useful.
- Rotated Images: some of these images have been derived from the regular blocks category, loaded onto a MS PowerPoint blank slide, rotated by a determined angle and then a screen shot of it has been taken and saved as a new image file to be tested as a rotated image.
- Other Languages: this is just an attempt to verify whether indeed this approach could be extended to applications involving languages other than Chinese, as is indeed the case.

A. Localized Otsu thresholding

The effect of this optional preprocessing step was a slight increase in the positive character hit ratio. The analogous adverse effect that could be observed in some cases is an increase in negative character hit ratio, where maybe non-Chinese text or other similarly repetitive structure was flagged as being part of the text, as seen in the below example:


iPod 9:47

风水人间不可无，也须阴骘两相扶。
时人不解苍天意，枉使身心着意图。话说
近代浙江衢州府，有一人，姓王名奉，哥
哥姓王，名春，弟兄各生一女：王春的女
儿名唤琼英，王奉的叫做琼真。琼英许配
本郡一个富家潘百万之子潘华。琼真许配
本郡萧别驾之子萧雅。都是自小聘定的。
琼英年方十岁，母亲先丧，父亲继殁。那
王春临终之时，将女儿琼英托与其弟，嘱
付道："我并无子嗣，只有此女，你把做
嫡女看成。待其长成，好好嫁去潘家。你
嫂嫂所遗房奁衣饰之类，尽数与之。有潘
家原聘财礼置下庄田，就把与他做脂粉之
费。莫负吾言！"嘱罢，气绝。殡葬事毕
，王奉将侄女琼英接回家中，与女儿琼真
作伴。

忽一年元旦，潘华和萧雅不约而同到
王奉家来拜年。那潘华生得粉脸朱唇，如
美女一般，人都称玉孩童。萧雅一脸麻子

页码 1/30 - 章节 2/41

a) Otsu OFF b) Otsu ON

Fig. 2. Comparison of character hit performace with local Otsu thresholding enabled (right) vs. disabled (left). For this case, the increase in postive hit rate was ~8%, but also had 4% more false hits (the light gray Western alphabet text at the bottom).

B. Fast Fourier Transform with radial projection

In order to simultaneously analyze the effectiveness of the FFT for finding the character repetition spacing and image angle of rotation, we selected one case where the image has in fact been rotated, as a representative example:

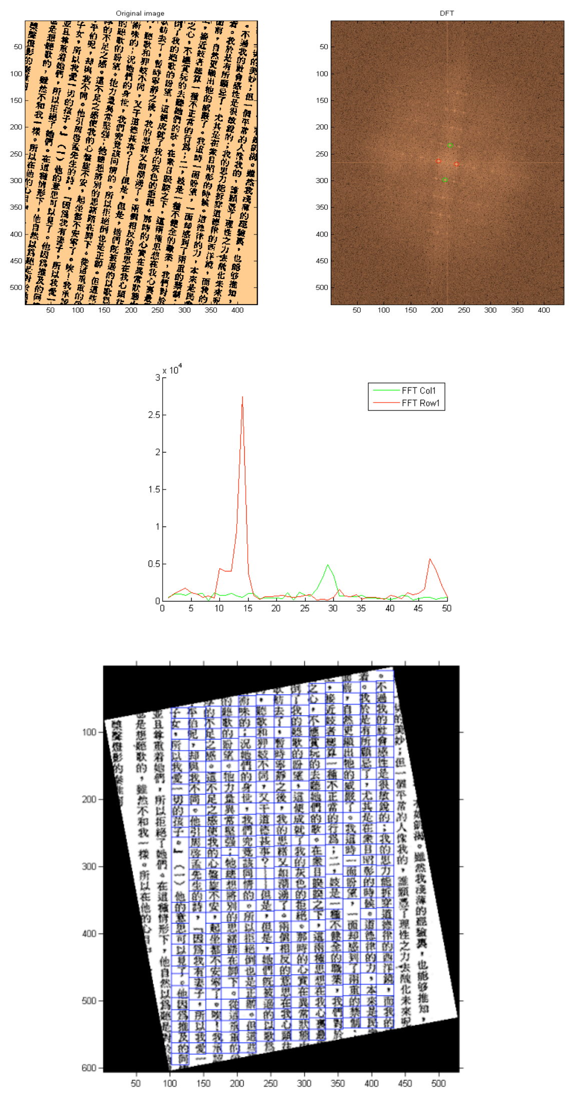


Fig. 3. Effectiveness of DFT method for both finding the primary peaks, as shown in the middle plot, and the image's angle of rotation. After rotating the image back to 0-90 degree orientation, the projection-based method can easily identify most of the characters in the image, despite some small residual angle still being present after the rotation.

We can get a rough idea of the accuracy of the FFT approach by comparing the line spacing delta as measured by this method as opposed to the subsequent and more accurate projection analysis (see Table I). As expected, the values are much closer for well-behaved cases, such as those containing regularly spaced blocks of text. However, even for well-behaved rotated images the FFT estimate will be quite close to the final projection estimate. For more irregular cases, the results fluctuate a lot from reasonably close cases, to some that are totally off (in other words, the FFT operation does not produce sufficiently clean peaks). One improvement that could be thought of here could be to try to apply a Gaussian filter directly to the FFT periodogram to see if more intense peaks could be achieved in such way.

TABLE I.

	Line spacing delta for DFT vs. projection (pixels)		
	Regular Blocks	***Irregular Blocks***	***Rotated Images***
Case1	0.344	1.432	0.390
Case2	0.007	1.911	1.233
Case3	0.198	0.139	N/A
Case4	0.105	15.720	
Case5	0.141	N/A	
Case6	0.128		
Case7	0.075		
Case8	0.091		
Case9	0.510		
Case10	0.527		
Average	0.21	4.80	0.81

Image projection profile analysis

The final processing step required for positive identification of spacing between character lines and characters within each text line is subdivided into two stages: the first stage involves doing a rough search for all the peaks in the projection profile. Incidentally, this peak search algorithm is identical to that used for finding the DFT peak, although two different criteria are used. These two criteria that the user can opt for finding peaks are: (1) finding outliers that are above the mean value by a certain factor of the standard deviation σ in the input data (default value: 2); (2) finding the total range of the data (defined as max-min values) and then keeping points that are within a factor of this range from the max value (default value: 0.15). It turns out that for the DFT peak finding, method (1) works best, while for projection peaks method (2) is preferable. The reason for this can be easily understood by comparing the characteristic signal of the two profiles, as seen in Fig. 4: the FFT profile has very sharp peaks that really stand out off a noisy low-lying background that dominates the mean and the σ value is very tight; the projection profile, in contrast, is in essence a bimodal distribution where the mean value is much closer to the peaks and the σ value is very large, thus the second approach tends to work more robustly. As a side note, the peak finder also takes care of eliminating peaks coming from blank (flat) areas at the image edges

The rough peaks found through this method are then sent to a second refinement stage consisting of a cascade of filters that include the following operations:

- Discard all peaks that are too close to each other (default is less than half of the DFT-predicted spacing). This takes care of discarding blank spaces and at least some of the punctuation symbols.
- Fill in gaps up to a certain size between peaks (default is up to 6 spaces), as long as the gap is close to an integer number of expected spaces (default is up to 15% deviation). This takes care of cases where the spacing between successive characters is very small. Some of these added peaks get cleaned up later.
- Outlier detection stage: compute mean and standard deviation of spacing between successive peaks found up to this point and discard any of them that satisfy the condition:

$$|\Delta| > \mu + \alpha\sigma \qquad (1)$$

(default α value of 2 has been used). When a peak is eliminated a new grouping of characters is created for later bookkeeping.

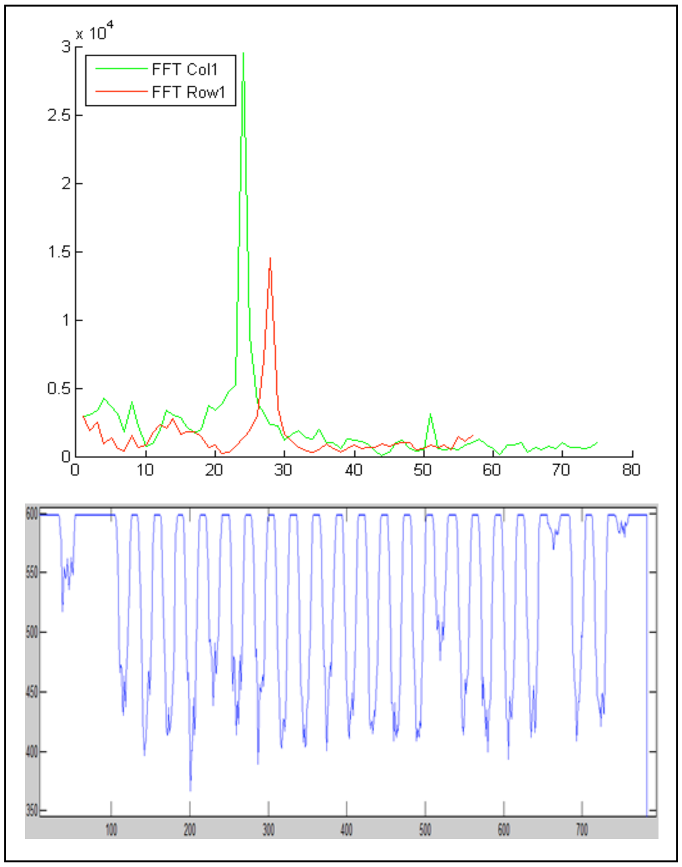


Fig. 4. Comparison of typical peak characteristics as seen in DFT profiles (top) as opposed to that obtained when taking direct image projections.

- For the remaining peaks, we look ahead at the contiguous spacing values and if they are in opposing trends (one is smaller and the other larger than expected), we rebalance the spacing values so that they are more equally distributed.

- The final step involves going through the different slices of the image thus found and eliminating those containing fewer than 3% of foreground pixels (roughly estimated by the mean intensity value, since the image has been normalized / binarized). This operation has the unfortunate drawback that it will also get rid of text lines with only a handful of characters (2-4) in them.

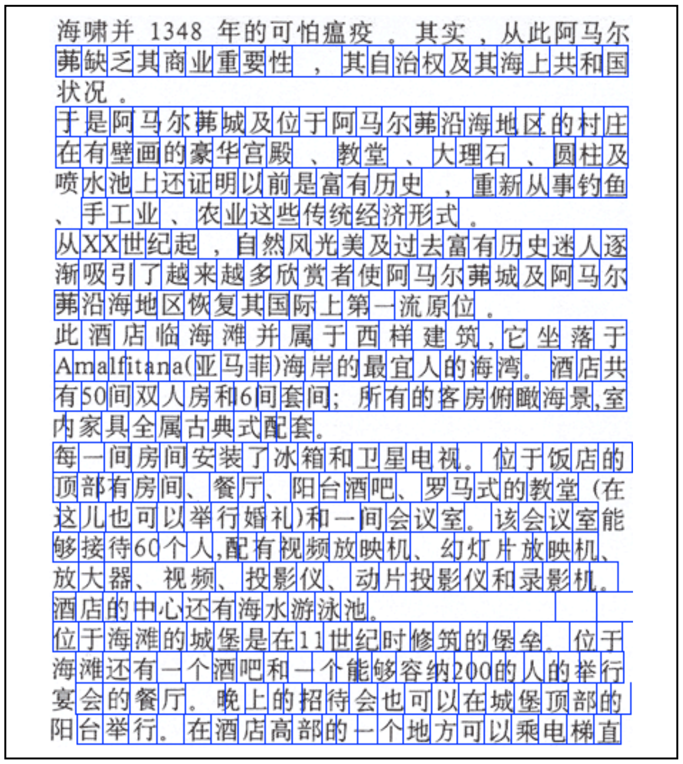

海啸并 1348 年的可怕瘟疫 。其实 ，从此阿马尔
菲缺乏其商业重要性 ， 其自治权及其海上共和国
状况 。
于是阿马尔菲城及位于阿马尔菲沿海地区的村庄
在有壁画的豪华宫殿 、教堂 、大理石 、圆柱及
喷水池上还证明以前是富有历史 ， 重新从事钓鱼
、手工业 、农业这些传统经济形式 。
从XX世纪起 ，自然风光美及过去富有历史迷人逐
渐吸引了越来越多欣赏者使阿马尔菲城及阿马尔
菲沿海地区恢复其国际上第一流原位 。
此酒店临海滩并属于西样建筑，它坐落于
Amalfitana(亚马菲)海岸的最宜人的海湾。酒店共
有50间双人房和6间套间；所有的客房俯瞰海景,室
内家具全属古典式配套。
每一间房间安装了冰箱和卫星电视。位于饭店的
顶部有房间、餐厅、阳台酒吧、罗马式的教堂（在
这儿也可以举行婚礼)和一间会议室。该会议室能
够接待60个人,配有视频放映机、幻灯片放映机、
放大器、视频、投影仪、动片投影仪和录影机。
酒店的中心还有海水游泳池。
位于海滩的城堡是在11世纪时修筑的堡垒。位于
海滩还有一个酒吧和一个能够容纳200的人的举行
宴会的餐厅。晚上的招待会也可以在城堡顶部的
阳台举行。在酒店高部的一个地方可以乘电梯直

Fig. 5. Example of performance of the proposed algorithm when applied to one of the Regular Block images. Most commonly observed issues: Missed hits (mostly at the image boundaries); Partial hits (symbols cut in half); False hits (extraneous non-Chinese symbols).

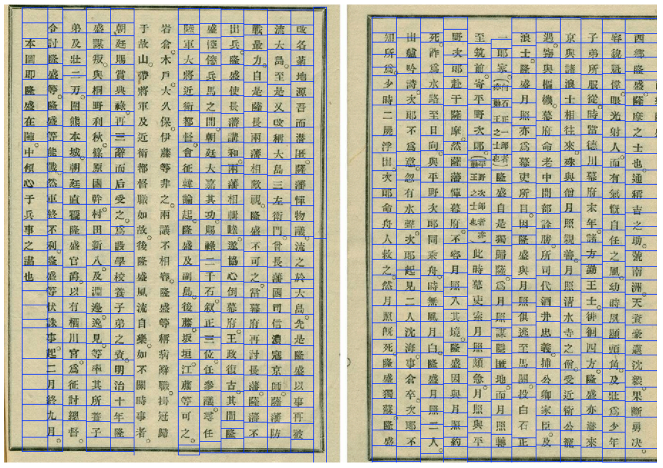


Fig. 6. Example of performance of the proposed algorithm when applied to one of the Irregular Block images. The issues remain the same, but their occurrence obviously increases, particularly for the Partial Hits category, as one would expect.

While this projection-based approach appears to be somewhat tedious and not so elegant, at the same time it produces quite reliable results under a variety of input conditions. Part of its beauty lies in the fact that all of the processing operations are linear and applied to profiles that are the length of one of the image sides, which means only a handful of numbers. Only the initial projection sums involve visiting all of the image pixels.

We have yet to make a thorough quantitative assessment of the accuracy of the application. Suffice it to say that for well-behaved cases (Regular Blocks) the positive hit ratio is over 90%, with maybe 5% missed characters; the rest is divided between partial hits, i.e. characters broken in half (an issue that can certainly be addressed by more specialized refinements); and false positives (non-Chinese symbols being flagged). For more ill-behaved ones (Irregular Blocks), the result quality will vary wildly, with somewhere between 60-80% positive hits, a large percentage of partial hits (due to the inherent irregularity) and quite a few missed hits also. Here one can experience first-hand the real limitations of a naïve projection-based approach. Considerable improvements would have to be made for this application to be able to handle these irregularities in a consistent fashion.

V. IMPLEMENTATION DETAILS

The application was implemented using Matlab. A special effort was made to limit the use of built-in functions (both in the general and Image Processing toolbox categories). The reason is that the code as written could be easily ported to a C/C++, as opposed to relying on analogous black boxes provided by Matlab. Where built-in functions were used, it is because analogous libraries exist in virtually any scientific programming language. These are mainly: *fft2*, *fftshift*, *imrotate*, *graythresh*, and *imftype*.

The code developed consists of roughly 1300 lines of code all told. It includes special testing capabilities to rotate any input image on-the-fly before being processed or to run a model image consisting of regularly spaced black boxes, where the sizes, spacing and phase can be arbitrarily specified.

The code also feature logging capabilities, encapsulating all of the useful quantities derived during the various image processing steps.

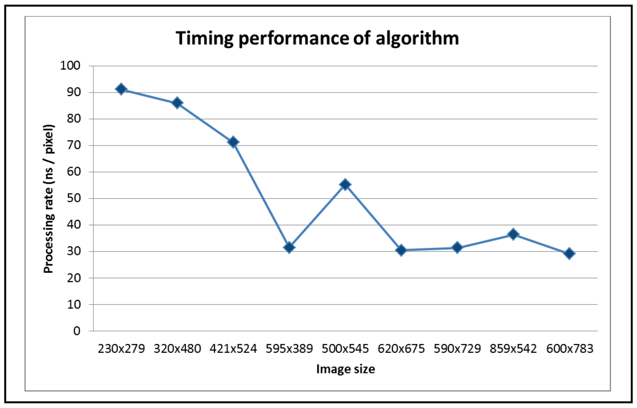


Fig. 7. Algorithm performance as rated in ns/pixels at increasing image sizes. As expected the performance increases, but plateaus at around 30ns/pixels.

The algorithmic performance rate, as measured on a 2.4GHz Intel Core2 Duo T8300 processor, was on average 50ns / pixel, but improves on larger image sizes with a plateau of roughly 30ns / pixels.

VI. FUTURE WORK

Specifically concerning this application, there are a few refinements needed to better handle (1) Missed Hits at the image boundaries; (2) Partial Hits by doing a second-pass through all of the character spacing values and requiring a minimum distance be satisfied; (3) False Hits, particularly punctuation marks and Western alphabet characters that may be interspersed in the text. However, the latter problem may be better handled during the back-end portion of the OCR processing. We believe addressing these problems is well within reach.

Beyond this there is the aspect of code optimization and streamlining of all the different operations. There is certainly room for improvement, especially relating to making the FFT even faster by not computing the coefficients for unneeded higher frequencies. Another aspect is to make most of the code really light weight so that some of the processing could run on a mobile architecture, such as an ARM processor, after the more demanding processing has been done on a server and results are returned.

What we would really have liked to do next is to finally harness the power of having all of the character bounding boxes and apply further image processing on them *individually*. This is very advantageous since the data is very small and the foreground pixels are now a majority. In principle, this should greatly simplify the segmentation of text pixels and more accurate results should be achievable.

Post segmentation, the idea would be to follow the rules of Chinese character decomposition, as explained in [8]. The vast majority of the characters fall under well-defined composition categories, each one having an established frequency of occurrence. With this information, one could analyze the segmentation output under all of the different composition kinds (in order of descending frequency) and selecting the one that fits the best. Based on this, the recognition problem is further reduced to identifying the character subcomponents, commonly known as “radicals”. Since there are less than 250 radicals that all the 20,000+ Chinese characters can be subdivided into, this makes the task of identifying the character meaning extremely simple, as long as radicals in the character can be identified with high confidence.

In order to achieve this, a database of radical image templates has to be constructed for comparing against. However, due to the large variety of fonts used, rather than using radical templates generated from an arbitrarily chosen font, one should come up with an abstraction model that the radical can be reduced to. If this can successfully be accomplished, admittedly a big “if”, then each radical can be reduced to a short and unique sequence of bits and the recognition task can be made lightning fast. The real challenge is to come up with rules for abstracting radicals, and strokes within them, in a robust manner.

Even without finding perfect matches, one can still assign a confidence score for each match and since only certain combinations of radicals are valid possibilities, by looking at all of the combinations with higher confidence the overall character matching confidence will increase dramatically.

The final piece of the puzzle would be to look at sequences of 2-3 characters and further weed out bad matches by looking at occurrence frequencies of such matches. When all of this is put together, the final outcome should be production of very high quality OCR of Chinese text.

VII. CONCLUSIONS

We have proposed in this paper a divide-and-conquer strategy for recognition of Chinese characters from images. As envisioned here, it is applicable primarily to images with arrangement of text in an orthogonal grid, as commonly found in books and newspapers.

The application developed in connection to this project relates to the front-end aspect of this problem, wherein robust identification of individual text symbols is a must. This is accomplished via a combination of the Fast Fourier Transform, followed by projection analysis methods and with minimal needs for image preprocessing.

The final outcome is a Matlab application that delivers according to initial expectations, with varying degree of output quality depending on how closely the input fits within the constraints of the model that the application was built around.

Several ideas are also put forth for future extension of this work as part of a more complete solution that accomplishes the overall text recognition goal, but strictly specialized to be applicable to the Chinese language.

REFERENCES

[1] Y. Lu and C. L. Tan, “Chinese Word Searching in Imaged Documents”, Int. J. Pattern Recognit. Artif. Intell., Vol. 18, No. 2 (2004), 229–246.

[2] Y. Liu, Printed Chinese Character Recognition, Hon. Sc. Thesis (http://www.massey.ac.nz/~albarcza/ResearchFiles/YuanLiu_Hon_2009.pdf) and references therein.

[3] R. Chen and S. Liao, “Mobile Chinese Translator”, EE368 Spring 2012 Project (http://www.stanford.edu/class/ee368/Project_12/index.html).

[4] T. Shioyama and J. Hamanaka, “Recognition Algorithm for Handprinted Chinese Characters by 2D-FFT”, 1996 IEEE Proc. of 13th ICPR, Vol.3, 225–229.

[5] E.F. Glynn, “Fourier Analysis and Image Processing”, online lecture, (http://research.stowers.org/microscopy/external/PowerpointPresentations/html/FourierAnalysis_files/frame.htm), 29 Sept 2006.

[6] J. F. Blinn, “What's that deal with the DCT?”, IEEE Comput. Graph. Appl., July 1993, 78-83.

[7] J. Liang, D. Doermann and H. Li, “Camera-based analysis of text and documents: a survey”, Intl. J. of Docum. Analysis and Recognit. (IJDAR), July 2005, Vol. 7, Issue 2-3, 84-104.

[8] WikiCommons Project, “Chinese characters decomposition” (http://commons.wikimedia.org/wiki/Commons%3aChinese_characters_decomposition).