Detecting Japanese characters from manga

Marten Hennoch
University of Tartu
marteff@ut.ee

ABSTRACT

Manga are comics created in Japan. The aim of this work is to create an application that allows users to detect Japanese characters from a picture of manga.

Keywords

OCR, Japanese, SIFT, manga

1. INTRODUCTION

Manga are comics created in Japan and they are very popular all over the world. Manga is usually black and white and the stories flow from top to bottom and from right to left. Nowadays a lot of manga is read online. Japanese language system consists of hiragana, katakana and kanji. While hiragana and katakana are quite easy to learn kanji are not. There are around 50 000 kanji but most of them are not used very often. Around 1000 - 2000 kanji will be enough for everyday use [19]. As there are very many different characters in the Japanese language it is quite hard for a non-native speaker to read it. Also most of the manga is in picture format so the reader cannot copy the characters and look for them in a dictionary. The aim of this work is to create an application that detects Japanese characters on a sheet of manga and allows users to copy them for translation. Existing open source libraries and tools for optical character recognition(OCR) are also explored. SIFT[11] is used for character detection .

2. EXISTING WORK

There has been some research on text recognition from comics and Japanese character recognition. In one of the few articles that focuses on manga the authors first extract frames from the page and then text bubbles that they then feed to AForge for OCR [1]. The authors of [4] use HOG as feature descriptor to find similar content in manga in order to detect forgings. The authors of [6] implemented a prototype of a mobile app for OCRing kanji. For feature extraction they use a gradient feature extraction method. The authors of [2] propose a clustering technique using a binary tree combined with CDA and compare the speed and accuracy with other kanji recognition techniques. [7] Authors propose two improvements to the active contour theory in order to detect open text balloons. But they expect that the text location is already known. The authors of [8] detect English text from comic book. The authors of [12] use SIFT to find Chinese text from google image search results.

There are many commercial applications that can OCR Japanese but not many free and open source ones. One free to use application is KanjiTomo[13]. When user hovers over a character it will OCR it and also offer a translation. Two open source OCR engines NHocr[9] and tesseract[10] were evaluated. NHocr can OCR machine printed horizontal text. Tesseract can OCR many languages but its Japanese support is very lacking.

3. IMPLEMENTATION

Scale-invariant feature transform (or SIFT) is an algorithm in computer vision to detect and describe local features in images. Initial idea was to implement SIFT in C++ and see it is possible to make it parallel in order to speed up keypoint detection because there are so many kanji. After reading the Lowes paper [11] and looking at the existing implementations it seemed that it would be good idea to first evaluate how well SIFT actually works using the implementation provided by openCV[14] and some other existing python libraries.

Keypoints are points of interest in an image. Keypoint usually holds only the information about its position. In order to compare the keypoints their descriptors need to be calculated also. Descriptor assigns a numerical description to the area of the image the keypoint refers to.

First step was to generate a collection of kanji images which keypoint detection could be run on. As different manga have different fonts function was needed that could draw kanji using a specific font. Using PIL(Python Imaging Library) author was able to draw an image with specific font and then convert it to openCV image. As SIFT should be scale invariant 100 x 100px kanji images were initially generated but testing indicated that using images that match the actual font size used in manga work better and provide more matches. After that keypoints were generated from all the kanji images and they were matched to given sample image which was an actual manga page sized 842 x 1200px. Then the matches were sorted by euclidean distance and 10 best matches for each kanji were saved for future processing.

Unfortunately there were too many false positives to match kanji to keypoints. At best each kanji had 2 matches. The main cause for this was the bad quality of the sample manga image. As most of them are scanned the text quality is far from machine printed text. In order to make kanji images resemble the test image kanji an averaging filter to blur the training images was used. This increased the amount of

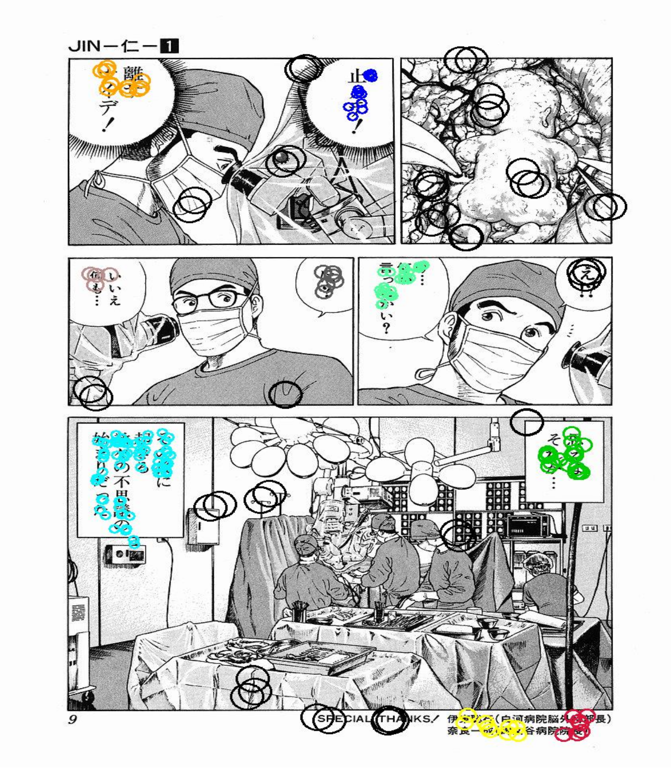

Figure 1: Sample manga[17] page with keypoints clustered by DBSCAN. Big circles are outliers.

matches by 5 %. Next the obvious outliers were removed. A keypoint was considered an outlier when it was 30px away from any other point. The SIFT algorithm itself had some parameters which could be tuned: contrastThreshold, edgeThreshold and sigma. For example by changing edgeThreshold value author was able to filter out edge-like features which were always going to be outliers. But even after all the parameter tuning there were too many false positives.

One idea to minimize the amount of false positives was to decrease the search space. Using the observation that most of the keypoints were usually in the chat bubble one could cluster the points in order to get the approximate area where the actual characters were located. At first k-means algorithm was tried for clustering but it gave bad results because k-means expects the amount of clusters being predefined and outliers affect the result quite a lot. Next idea was to use some density based algorithm because the most dense regions which would give the most accurate text bubble location were needed. DBSCAN[15] algorithm was used. The results were quite good Figure 1. After that some coordinates from the cluster keypoints were calculated in order to crop out a rectangular area in which the text bubble resides. Figure 2.

Next keypoint detection and matching was run on smaller text bubble images. If some configurable amount of keypoints were in some radius of each other that location was marked as a kanji. This approach worked quite well on most of the kanji. Not very well on hiragana and katakana because they are simple characters and don't have many keypoints to begin with. But they are not imporant anyways because most of the people who would want to read kanji can read them already. After adding more characters to kanji library this approach unfortunately became quite slow. The slowness was mostly due to the matching taking 80 seconds with 2000 kanji.

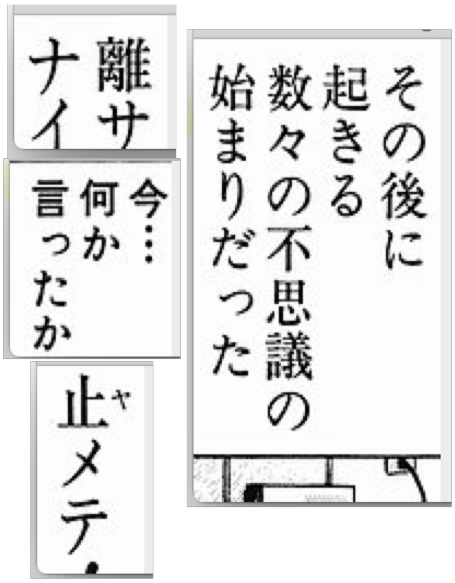

Figure 2: Detected text areas

4. CONCLUSIONS

In conclusion I can say that SIFT seems quite viable when detecting rough kanji location. Problems arise when trying to match kanji to kanji. Main problem is that training images don't seem to match very well with actual manga scans. There are several things that can be done to improve it. I tried blurring the images and some parameter tuning. Author also only tried generating training images with one font only. So there are many small adjustments that can be made in order to make this approach better.

There also exists a SIFT for GPU which could be used to generate large amount of keypoints faster. In future I'll try to implement similar application in C++ and see how can I use GPU or parallelization to speed things up. Another thing I want to try is to use ORB[16] instead of SIFT. SIFT is patented but ORB has BSD license.

5. REFERENCES

[1] Koga, M.; Mine, R.; Kameyama, T.; Takahashi, T.; Yamazaki, M.; Yamaguchi, T., "Camera-based Kanji OCR for mobile-phones: practical issues," Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on , vol., no., pp.635,639 Vol. 2, 29 Aug.-1 Sept. 2005

[2] Y. Sobu, H. Goto, "Binary Tree-Based Accuracy-Keeping Clustering Using CDA for Very Fast Japanese Character Recognition". MVA2011 IAPR Conference on Machine Vision Applications, June 13-15, 2011, Nara, JAPAN

[3] Yi Zhou; Kai Chen; Xiaokang Yang, "Google image

search refinement: Finding text in images using local features," Computer Science and Automation Engineering (CSAE), 2012 IEEE International Conference on , vol.1, no., pp.98,101, 25-27 May 2012

[4] Kohei, A., Tolle, "Method for Real Time Text Extraction of Digital Manga Comic", International Journal of Image Processing 2011

[5] Weihan Sun; Kise, K., "Similar Manga Retrieval Using Visual Vocabulary Based on Regions of Interest," Document Analysis and Recognition (ICDAR), 2011 International Conference on , vol., no., pp.1075,1079, 18-21 Sept. 2011

[6] Koga, M.; Mine, R.; Kameyama, T.; Takahashi, T.; Yamazaki, M.; Yamaguchi, T., "Camera-based Kanji OCR for mobile-phones: practical issues," Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on , vol., no., pp.635,639 Vol. 2, 29 Aug.-1 Sept. 2005

[7] Rigaud, C.; Burie, J.-C.; Ogier, J.-M.; Karatzas, D.; van de Weijer, J., "An Active Contour Model for Speech Balloon Detection in Comics," Document Analysis and Recognition (ICDAR), 2013 12th International Conference on , vol., no., pp.1240,1244, 25-28 Aug. 2013

[8] Christophe Rigaud, Dimosthenis Karatzas, Joost Van De Weijer, Jean-Christophe Burie, JeanMarc Ogier. Automatic text localisation in scanned comic books. 9th International Conference on Computer Vision Theory and Applications, Feb 2013, Barcelona, Spain.

[9] http://sourceforge.jp/projects/nhocr/

[10] https://code.google.com/p/tesseract-ocr/

[11] David G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, 60, 2 (2004), pp. 91-110.

[12] Yi Zhou; Kai Chen; Xiaokang Yang, "Google image search refinement: Finding text in images using local features," Computer Science and Automation Engineering (CSAE), 2012 IEEE International Conference on , vol.1, no., pp.98,101, 25-27 May 2012

[13] http://kanjitomo.net/

[14] http://opencv.org/

[15] http://en.wikipedia.org/wiki/DBSCAN

[16] Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G., "ORB: An efficient alternative to SIFT or SURF," Computer Vision (ICCV), 2011 IEEE International Conference on , vol., no., pp.2564,2571, 6-13 Nov. 2011 doi: 10.1109/ICCV.2011.6126544

[17] Dr Jin vol 1. by Motoka Murakami

[18] https://bitbucket.org/marteffyo/hs2014

[19] http://en.wikipedia.org/wiki/Kanji#Total_number_of_kanji From: 16.12.2014