National Institute of Informatics, Tokyo
Internship Report

Detection of Differences between Printed Pages and its Application on Bukan

Thomas Leyh
University of Freiburg, Germany

March 20th, 2020

Table of Contents

1 Introduction

The *Center for Open Data in the Humanities* (CODH) is a joint research institution in Tokyo, Japan. Its goal is the promotion of data-driven research in the humanities.[1] For this purpose they were releasing a number of data sets, all of them related to Japanese history and arts. This work started out by specifically looking at the *Bukan Collection*[2], around 370 books with information about government officials during the 18th and 19th century. So, how can we assist humanities researchers with techniques from Computer Science?

By applying well-known algorithms from Computer Vision on the books' pages, without using information about the written content, a reliable system for spotting and visualizing similarities has been developed. This might be a first step into building a timeline of a book's prints, thus exposing some historical events hidden in there.

But more importantly, this is an example of how even conservative algorithms can give compelling results by using a few reasonable assumptions on the corresponding data. Even basic computer-aided quantitative analysis might reveal information that is near invisible to a human researcher.

The full source code of this work can be found at: `https://github.com/leyhline/bukan-collection/`

2 The Bukan Collection

This work was mainly concerned with extracting information from a specific type of book: 武鑑—Bukan. These are historic Japanese books from Edo period (1603-1868). They are about listing people of influence, i.e. persons, families and institutions with governmental responsibilities. Alongside names, there are also various symbols like family crests and procession items as well as family trees. The books are written in pre-modern Japanese (*Kuzushiji*), therefore they can not easily be read without special training.[3] See figure 1 for an example.

Back then, the Bukan were bestsellers. Often, it was vital knowledge to be able to identify feudal lords and their subordinates. Mistakes when determining the difference in status could easily lead to dispute, sometimes even in a violent manner.[4] Nevertheless, publishing of the Bukan was not centralized and therefore not standardized. Especially the layout of the pages—even of books from the same year—might differ to some degree.

The language barrier and the layout differences might seem to complicate the application of common algorithms from Natural Language Processing

Figure 1: Shūgyoku Bukan (袖玉武鑑) from 1861, page 6; showing names, descriptions, family crests and procession items. Especially interesting are the blank areas on the right. These might be filled in later prints.

and Computer Vision. But utilizing the knowledge about the specific data at hand, about the Bukan and especially about their production, defining a more approachable problem is possible. This can be done by looking at the printing process.

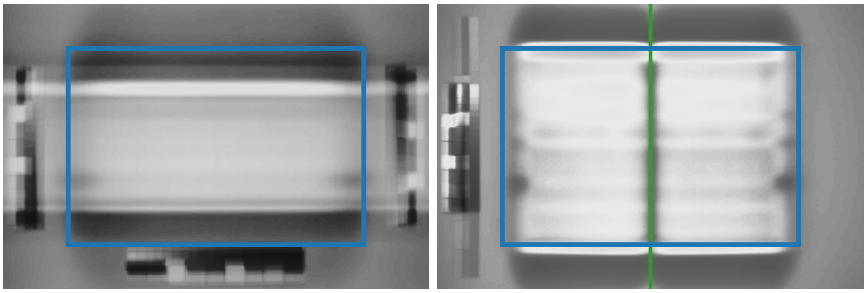
The books were created using Japanese woodblock-printing. Specialized craftspeople were carving out whole pages from wood. By applying ink on these blocks, numerous prints were created with comparatively low costs per unit. But it is not like all pages made with the same woodblock would look identical. On the one hand, there are differences in quality of the printed page, stemming from e.g. the amount of ink or the state of the woodblock. On the other hand, the woodblocks itself might have been modified between prints. Some names could have been added, some removed or some symbols modified.

These differences might indicate historical occurrences: changing family structures like births and deaths as well as gain or loss of status. A scholar in history and literature will be able to interpret these findings. This work is about supplying her with interactive tools for quickly spotting such sections of interest.

3 Method

For tackling this task, three common techniques are used:

1. *Keypoint Detection and Matching*, a method from Computer Vision



(a) (b)

Figure 2: The mean over all grayscaled images grouped by book layout: (a) shows the mean of scans with wide, single pages while (b) shows images with two pages per scan. The blue rectangle shows the cropping window and the green line represents the horizontal center of the scan. It seems to correspond with the binding, therefore averagely separating left from right page.

used for finding the same object in different images.

2. *Projective Transformations* for comparing two different images, regardless their original orientation.

3. *Relational Databases* for easily storing results as well as querying for relevant pages.

At the end there will be a database populated with the necessary data for easily browsing books, finding similar pages and quickly displaying visualizations. For all these steps free and open tools are available. Especially for the Computer Vision parts the mature *OpenCV* software library was used.[5]

3.1 Keypoint Matching

As a first step, it is necessary to compare a large number of images. This must be computationally viable and invariant to a number of image transformations. The original images—as they were stored by the digital camera—have a width of 5616 and height of 3744 pixels. Thus, each image has around 21 million pixels with three color channels each.

Under the assumption that this task (1) does not require this level of detail, (2) does not require information about color and (3) only compares the actual pages, not the surrounding area, basic image transformations are applied. All scans are resized by 25%, converted to grayscale and finally

cropped, resulting in an image shape of 990×660 pixels. If there are two book pages per scan, they were additionally split at their horizontal center. See figure 2.

Now, each image has still around $650,000$ pixels. Moreover, at least some *translational invariance* is required since the images are not always perfectly aligned to each other. It is also necessary to consider that letters and symbols vary in thickness, depending on the amount of ink used during printing. This is where *Keypoint Detection* comes in. The general idea is to find points of interest in an image that are most noticeable and give a unique description of the local area surrounding them. This description is mostly abstract, hence invariant to many kinds of transformations. For details, see [6, Ch.4]

Computer Vision research produced various kinds of keypoints, most prominently *SIFT*.[7] For evaluating the performance of these algorithms, 12 prints of the *Shūchin Bukan* (袖珍武鑑) were manually annotated, in total around 1800 pages, holding information about the position of matching pages. However, most of the time an identical page number corresponds to matching page content (e.g. page 10 of two prints was made using the same woodblock). To remove this bias, a potential matching of all possible page-keypoint combinations was computed while discarding matching pages where the number of matching keypoints was below a given threshold. The metric for evaluation is the value at the intersection of the precision and the recall curve. The following algorithms were evaluated:

- ORB[8]
- AKAZE[9]
- AKAZE without rotational invariance (UPRIGHT)
- BRISK[10]
- SIFT and SURF[11], but these are not further considered since they are lacking performance on this task

The results are presented in figure 3. AKAZE UPRIGHT seems to have the best performance. Here, rotational invariance was explicitly *not* included since the original images were scanned after cleanly aligning them on a table. This seems to bolster the keypoints' discriminative power. Nevertheless, the current results are far from satisfactory; from a performance standpoint as well as from the stability of the approach. Can we improve upon this by incorporating additional information?

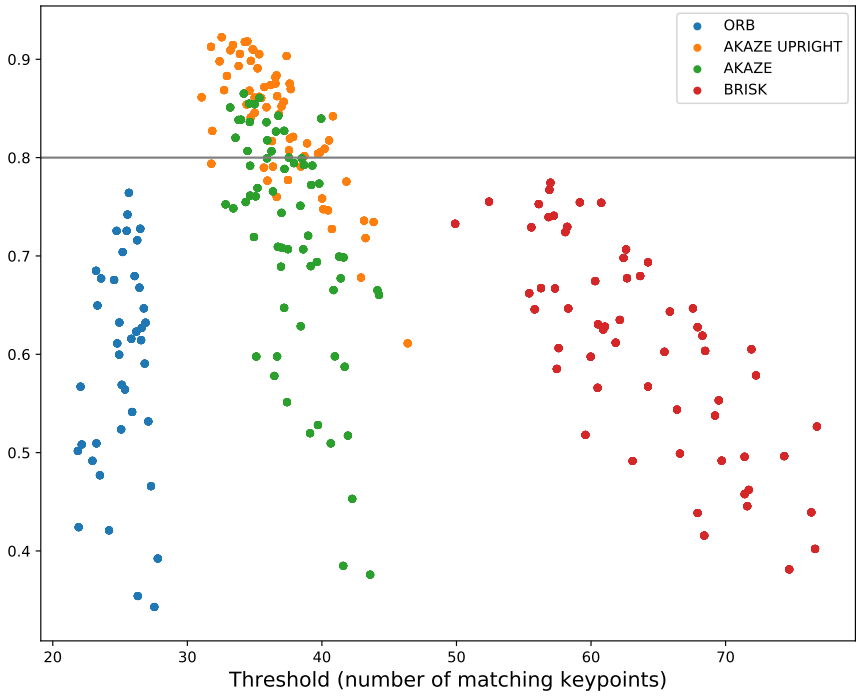


Figure 3: Scatterplot showing the positions of intersection of various precision-recall curves depending on the threshold of the number of matching keypoints. All points above the gray horizontal line at 0.8 are considered "good enough" by the author. Even though this metric discards much information, there is a general tendency in favor of AKAZE UPRIGHT discernible.

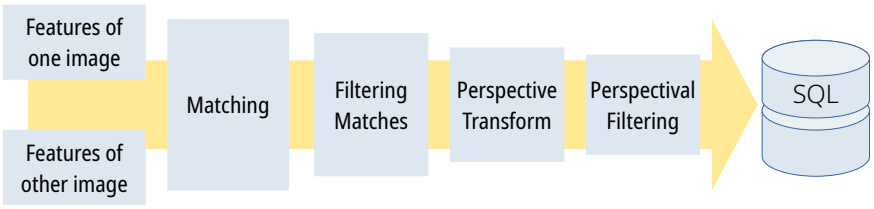


Figure 4: Starting out from Keypoint Matching described in section 3.1 a pipeline was built, reducing the number of false positives by incorporating additional information and storing the results for further usage in a relational database (SQL).

3.2 Projective Transformations

There is, of course, more information that can be used for improving the performance. As a first step, filtering matching keypoints by their quality and position is useful. Here, matches with a distance[1] above 100 are removed. The same goes for matches, where the keypoint pair is more than 100 pixels apart from each other. With this, larger translations are not allowed since we assume clean alignment during the scanning process. But by far the largest improvements in performance are gained by attempting a *Projective Transformation* between two images.

A Projective Transformation (or *Homography*) is a matrix $\mathbf{H}$ for transforming homogeneous coordinates $\mathbf{H}\vec{x} = \vec{y}$. This not only allows linear transformations like translation and rotation but also perspective transformations. Here—since the task is about flat images—the matrix is three-dimensional: $\mathbf{H} \in \mathbb{R}^{3\times3}$. Details can be found in most books on Computer Graphics like [12] since these calculations are elementary to modern 3D applications. For finding such a transformation from matching keypoints—even though it is not sure if these matches are always correct—the heuristic *Random Sample Consensus* (RANSAC) algorithm is used.[13] Its basic concept is that a random subset of matching keypoint pairs is chosen, using this to compute a transformation and calculating an error metric. By iteratively using different random subsets, eventually the transformation with the smallest error is picked.

Using RANSAC leads to outstanding results. Values for precision and recall—depending on the threshold set for the number of keypoints—are close to their maximum of 1.0. Moreover, the performance is much more

[1] AKAZE keypoints yield binary descriptors. Two such descriptors can be compared calculating the *Hamming Distance*, where larger distances implies a larger perceptible difference.

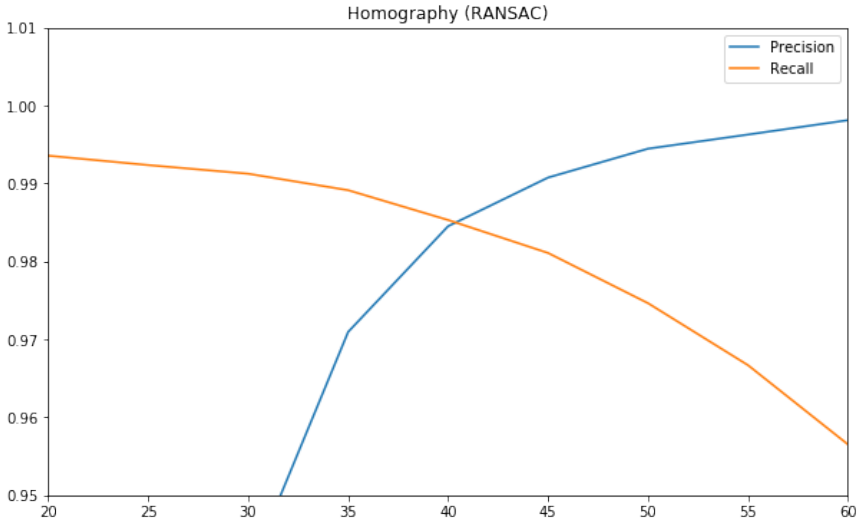


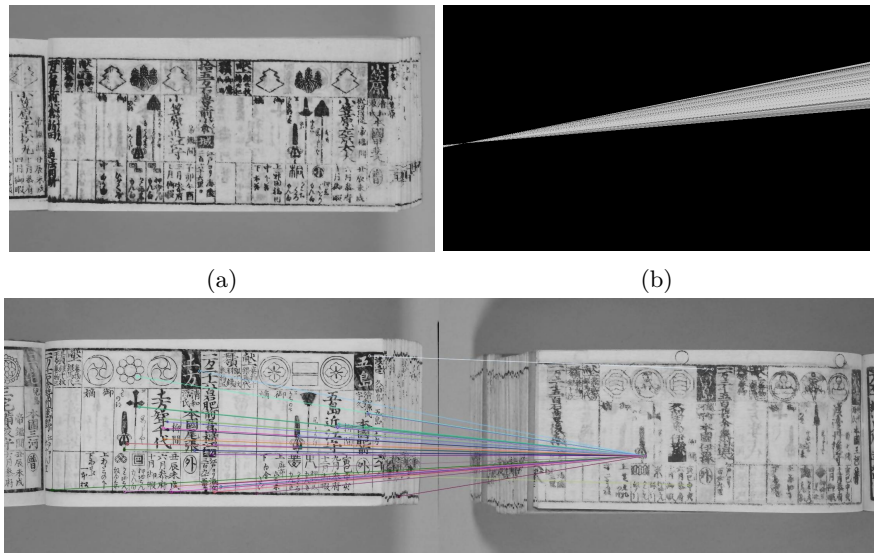
Figure 5: Precision and recall, by threshold on the number of keypoints. This is but a small part from the whole curve, just showing values above 0.95. The recall is always on an excellent level while the threshold should not be too low because precision will collapse otherwise.

robust concerning this threshold. See figure 5 where even for larger thresholds the recall is always above 0.95. These results come at a cost: a computational one. Calculating RANSAC is much more expensive in this regard than just matching keypoints. Of course, first filtering the keypoints before running RANSAC is an option for speeding up the computation, but too aggressive filtering will go at the expense of robustness. After some experiments, a maximum distiance between keypoints of 100 was chosen.

A last source of failure that can easily be eliminated are extreme transformations in perspective as seen in figure 6. Caused by incorrect keypoint matching, RANSAC mistakenly results in extreme perspective shifts that nevertheless are deemed possible by the algorithm. This makes up for less than 1% of false positives, but can be corrected by removing matches where the projective elements from $\mathbf{H}$ are too extreme. Accordingly, transformations must conform to these two inequalities:

$$|\mathbf{H}_{1,3}| \leq 0.001$$
$$|\mathbf{H}_{2,3}| \leq 0.001$$

The threshold of 0.001 is quite conservative and might even be chosen one magnitude lower without much difference.



(a) (b)

(c)

Figure 6: Two examples for projective transformations: In this case, a good transformation as in (a) will have no perceptible effect, just slightly shifting some pixels. The image in (b) is the result of a bad transformation, shifting perspective in a way the original image can not be recognized anymore. This is caused by incorrect keypoint matching as seen in (c) where the majority of keypoints on the left page is matched to a single point on the right page.

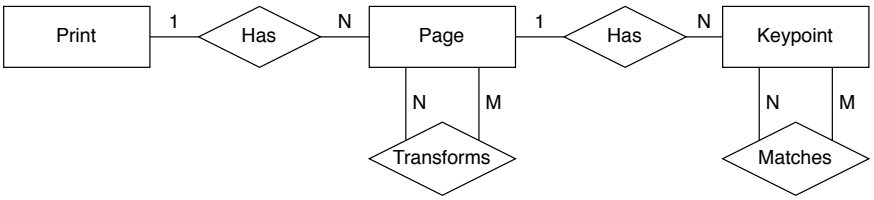


Figure 7: An entity-relationship diagram describing the relations between data points; e.g. each print has multiple pages, each page has multiple keypoints, and keypoint can potentially be matched to each other.

3.3 Relational Databases

There are numerous possibilities for storing the results from the aforementioned calculations (keypoints, matches, transformations). Here, it seems most sensible to use relational databases, a mature concept based on Codd's *Relational Model* from 1970.[14] This allows for easy preservation of the relationships between the data points as well as easy retrieval.

The entity-relationship diagram in figure 7 can give a rough overview of the general structure of the database. There are numerous software implementations available and most frontend frameworks offer bindings, thus allowing for easily building visualizations. When inspecting these relations in detail, another problem occurs; again of computational nature. Reading the diagram from left to right:

- Each print has around 250 pages on average,
- each page has around 300 keypoints on average,
- and naive matching requires to look at every possible keypoint pair combination.

Hence, for 370 books, the total number of keypoints is approximately $n = 370 \cdot 250 \cdot 300 = 27,750,000$. The number of combinations is given by:

$$\binom{n}{2} = \frac{n!}{2!(n-2)!} \approx 3.85 \times 10^{14}$$

Even using the abundant computational power that is available nowadays, calculating 400 trillion matches for just a few hundred books is infeasible, not to mention storing them. It might be possible to speed this up algorithmically, using suitable data structures. Instead, a simpler approach was chosen.

By making two assumptions about the data—the Bukan Collection—it is easy to drastically reduce the number of matching operations. First,

matching pages are only expected between prints of the same book title.[2] Second, the offset between pages should not be more than 8 pages in both directions; for example page 10 of one book will not be compared to all the pages of the other book anymore. Instead, this page 10 is only matched with pages 2-18 (in total 17) of the other book. For two books with 250 pages this would result in much less page comparisons:

$$\begin{aligned} 250^2 &= 62,500 && \text{naive matching} \\ 250 \cdot 17 &= 4,250 && \text{matching after fixing page offset} \\ \frac{62,500}{4,250} &\approx 14.7 && \text{speedup factor} \end{aligned}$$

Moreover, from a complexity standpoint, there is no quadratic increase in the number of matches anymore. Instead, it is increasing linearly, leading to an even larger "speedup factor" for books with more pages.

4 Conclusion

Even though many techniques were applied, it is primarily just about keypoint matching and running RANSAC on the results. Both are mature, well-understood and most importantly well-documented. Likewise, there are many parameters when setting up this pipeline. But it seems like the majority does not have a significant influence on the overall performance (save for the threshold at the very end of the pipeline), thus allowing easy implementation.

At the end of this work, there is a database holding information about matching keypoints and page transformations. For each matching page pair found, it is now possible to align them and applying simple, pixelwise operations for visualizing the differences. For an example see figure 8.

A scholar in the humanities and literature can now interactively browse these books and quickly identify interesting parts. This will hopefully lessen her burden when examining numerous pages, since small, abstract differences are often hard to discern for the human eye. So instead of exhausting her cognitive ability by staring at pages alone, it will now hopefully be easier to extract information from the actual content.

[2] There are 44 distinct book titles in the Bukan Collection, each with a different number of prints, ranging from 100 prints for the Shūgyoku Bukan (袖玉武鑑) down to only a single print for 14 titles.

Figure 8: Visualizing page differences between two prints of the Shūchin Bukan (袖珍武鑑); page 13 from 1840 and page 15 from 1837. Differences are indicated by blueish and reddish coloring respectively.

5 Further Improvements

Naturally, multiple improvements are possible—most prominently runtime optimizations—that open up at every stage of the pipeline. Hereafter is a non-conclusive list of starting points:

During keypoint matching (section 3.1) better strategies for removing problematic keypoints and matches are possible.

RANSAC (section 3.2) has multiple parameters that can be tuned. Furthermore, there even might be better, non-iterative methods for finding correct keypoint pairs for the problem and data at hand.

As already mentioned in section 3.3, there might be useful data structures for speeding up the search for matching keypoints (i.e. computing the *hamming distance*). Furthermore, the assumptions made on the data might need further refinement. Especially grouping prints by title might lead to interesting page pairs going undetected. Grouping by year published might deserve investigation.

A most interesting improvement would be a method for reliably flattening out pages before running the pipeline. Since the historic books presented here were digitized with much care, most pages are curved near their binding. Otherwise, books would have been damaged during the scanning process. Having images of completely flat pages would certainly improve keypoint matching as well as visualization. Some effort was made in this direction, but nothing proved stable enough, thus hurting the overall performance.

References

[1] Asanobu KITAMOTO. "Center for Open Data in the Humanities (CODH): Activities and Future Plans". In: *1st CODH Seminar: Big Data and Digital Humanities*. Jan. 2017. DOI: 10.20676/00000001.

[2] Center for Open Data in the Humanities. 武鑑全集とは？. July 28, 2018. URL: http://codh.rois.ac.jp/bukan/about/.

[3] Hakim Invernizzi. "An iconography-based approach to named entities indexing in digitized book collections". MA thesis. École polytechnique fédérale de Lausanne, Sept. 2019.

[4] John W Dower. *The elements of Japanese design: a handbook of family crests, heraldry & symbolism*. Weatherhill, Incorporated, 1990.

[5] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[6] R. Szeliski. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. Springer London, 2010. ISBN: 9781848829350.

[7] David G Lowe. "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2 (2004), pp. 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94.

[8] E. Rublee et al. "ORB: An efficient alternative to SIFT or SURF". In: *2011 International Conference on Computer Vision*. Nov. 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

[9] Pablo F Alcantarilla and T Solutions. "Fast explicit diffusion for accelerated features in nonlinear scale spaces". In: *IEEE Trans. Patt. Anal. Mach. Intell* 34.7 (2011), pp. 1281–1298. DOI: 10.5244/C.27.13.

[10] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. "BRISK: Binary robust invariant scalable keypoints". In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2548–2555. DOI: 10.1109/ICCV.2011.6126542.

[11] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features". In: *European conference on computer vision*. Springer. 2006, pp. 404–417. DOI: 10.1007/11744023_32.

[12] Steve Marschner and Peter Shirley. *Fundamentals of computer graphics*. CRC Press, 2015.

[13] Martin A Fischler and Robert C Bolles. “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”. In: *Communications of the ACM* 24.6 (1981), pp. 381–395. DOI: 10.1145/358669.358692.

[14] E. F. Codd. “A Relational Model of Data for Large Shared Data Banks”. In: *Commun. ACM* 13.6 (June 1970), pp. 377–387. ISSN: 0001-0782. DOI: 10.1145/362384.362685.