A report on Image Processing Course (INT 3404)
Word Segmentation with Scale Space Technique
(based on the research of Manmatha and Rothfeder)

Viet-Hoang Nguyen, Quoc-Hung Duong, Duy-Long Nguyen
VNU University of Engineering and Technology, Hanoi, Vietnam.
`{18020504, 18020021, 18020790}@vnu.edu.vn`

Abstract

This document is a short report generalizing ideas of the paper named *Word Segmentation with Scale Space Technique* (Manmatha and Rothfeder, 2005). Optical Character Recognition (OCR) is one of the earliest computer vision tasks, developed by many researchers with various approaches. The important steps in this task are segmentation the document page into words and grouping instances of the same word by using image matching. Back in 2005, when the Machine Learning- and Deep Learning-based methods were not developed, authors of the given paper proposed an algorithm based on image processing techniques called Scale Space Technique.

.

1 Introduction

There are numerous single-author historical handwritten manuscripts that could be indexed and searched. In the year 2005, most of these manuscripts were created page by page manually. Therefore, detecting words in pages automatically would be useful and was taken into consideration. In the given paper, the authors come up with an algorithm for word segmentation in images of historical documents by considering the scale space behavior of ink stain line by line in the document pages.

At that time, no good enough techniques existed to segment words of such handwritten manuscripts. Moreover, the scale space technique was not applied into this problem before. Those were the motivation that promoted the author to follow this task.

There are some challenges that can be specified in the historical handwritten manuscripts' word segmentation task: **(i)** most of the documents suffered from various problems including noise, shine through and other artifacts due to aging and degradation and **(ii)** handwritten documents are variant due to writing styles, inconsistency of word-to-word and line-to-line distances. Those challenges were also considered by the authors in this paper.

2 Related work

Before proposing this paper, the authors also developed another method to segment words called *Word Spotting* (Manmatha et al., 1996), which mainly only focus on word matching strategies and did not address full-page segmentation issues in handwritten documents. With the incredible development in Machine Learning and Deep Learning, many approaches with high performance are introduced, for example, *Word Spotting in Handwritten Manuscripts by using the Deep neural network called Ctrl-F-Net* (Wilkinson et al., 2017).

Even though the approach of the authors is outdated and not good enough compared with current novel approaches, in the year 2005, it could support people to reduce manual works of indexing handwritten documents, which significantly cost much time.

3 Materials & Algorithms

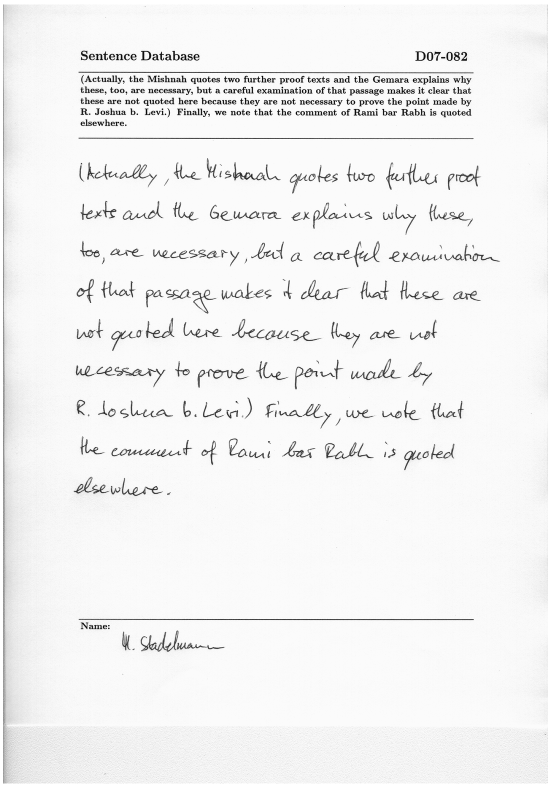
3.1 Datasets

To train and evaluate the approach of the given paper, we decided to use the *IAM Handwriting Dataset* (Marti and Bunke, 2002). Some statistical data of the dataset are shown in the **Table 1**.

Table 1: Statistics of the *IAM Handwriting Dataset*.

Statistic aspect	Value
Writers	657
Images (scanned from handwritten manuscripts)	1.539
Sentences	5.685
Lines	15.353
Words	115.320

Note that the images are scanned at a resolution of 300 dpi and saved in PNG format with 256 gray levels. The dataset consists of solely handwritten English text and all the sentences, lines and words are labeled so it can be conveniently used for training and evaluating purposes. Moreover, dataset also includes information about identifies of many people who are authors of documents. An example of the input is shown in **Figure 1** below.



Sentence Database D07-082

(Actually, the Mishnah quotes two further proof texts and the Gemara explains why these, too, are necessary, but a careful examination of that passage makes it clear that these are not quoted here because they are not necessary to prove the point made by R. Joshua b. Levi.) Finally, we note that the comment of Rami bar Rabh is quoted elsewhere.

(Actually, the Mishnah quotes two further proof
texts and the Gemara explains why these,
too, are necessary, but a careful examination
of that passage makes it clear that these are
not quoted here because they are not
necessary to prove the point made by
R. Joshua b. Levi.) Finally, we note that
the comment of Rami bar Rabh is quoted
elsewhere.

Name: M. Stadelmann

Figure 1: Example sheet in the *IAM Handwriting Dataset*.

3.2 Proposed algorithms

In this report, we briefly describe the general idea of the given paper with some following steps:

1. An input gray-scale document image is preprocessed to remove horizontal and vertical lines which can interfere with later operations.

2. The page is separated into lines by using projection analysis techniques with some modifications for gray-scale image, with the projection function is smoothed by low-pass filtering to eliminate false alarm and detect white space between lines.

3. Each line is then convolved with the second-order anisotropic Gaussian derivative filter to create a scale space to give attention to words in the document, the scale factor is selected automatically by an efficient heuristic function.

4. Finally, words are extracted by blob-connected components analysis followed by generating bounding boxes as the output.

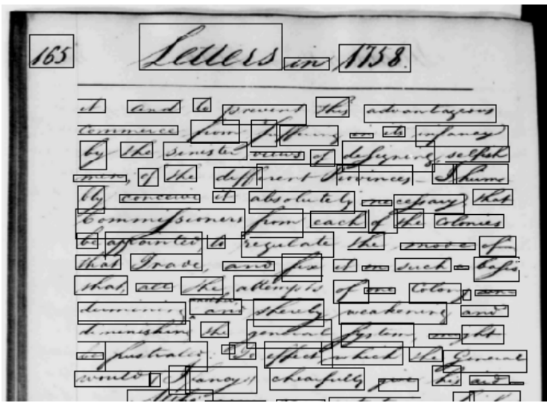


Figure 2: Example of the words segmentation results on the George Washington handwritten manuscript.

3.3 Detail algorithms

We have described the general idea briefly at **Section 3.2**, so we are going to get into the detail of it. In the authors' work, they use images of handwriting documents as input. We can consider a document as a combination of features at multiple scales. More specifically, at one scale, we have characters, and at higher scales, we have words, phrases, lines, and so on. In the case of our work, we want to focus on word as a feature. Hence, we would like to have an image representation with a word-level scale in order to derive features accurately, which makes use of *Linear scale space representation* (Lindeberg, 1994).

To form a linear scale space representation of a image, we use the Gaussian filter. The linear scale space representation of a continuous signal with arbitrary dimensions consists of building a one-parameter family of signals derived from the original one in which the details are progressively removed. Let $f : \mathbb{R}^2 \to \mathbb{R}$ represent a given signal and $I : \mathbb{R}^2 \times \mathbb{R}_+ \to \mathbb{R}$ is the scale space representation. We define I by letting the scale space representation at zero scale be equal to the original signal $I(\cdot; 0) = f$ and for $t > 0$:

$$I(\cdot; t) = G(\cdot; t) * f \qquad (1)$$

where $t \in \mathbb{R}_+$ is the scale parameter and G is the Gaussian kernel. The Gaussian kernel in two dimensions ($x, y \in \mathbb{R}$) is written as:

$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{(2\sigma^2)}} \tag{2}$$

where $\sigma = \sqrt{2t}$.

We now describe the details of the authors' algorithm. The **Figure 3** show the flow of how the algorithm works.

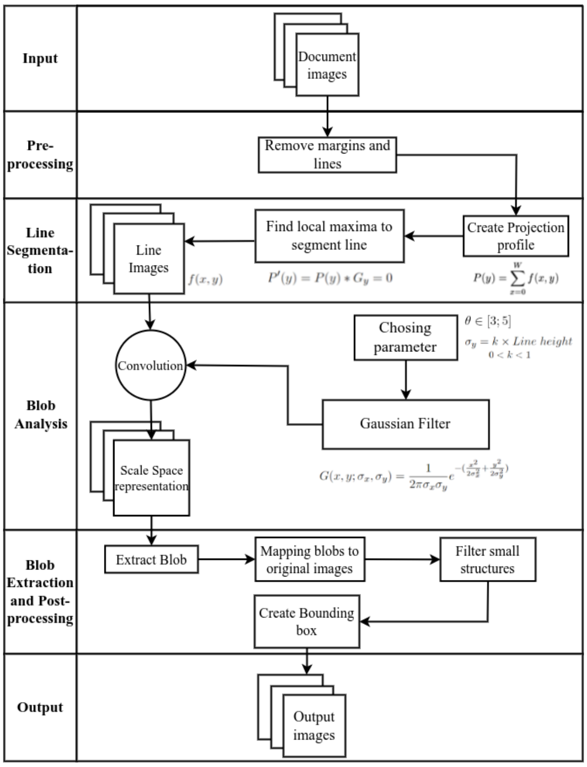


Figure 3: The flow of the algorithm.

3.3.1 Pre-processing

Due to the characteristic of the handwritten manuscripts, the input may suffer from degradation. Furthermore, the inputs were obtained by scanning the photocopies of original manuscripts. This process may create some horizontal and vertical black line segments or margins which affect the later work. Therefore, in this stage, the authors need to remove some of these margins and lines. The pre-processing step is described in the research of Srimal.

3.3.2 Line Segmentation

We can see that the lines in the manuscripts consist of a succession of horizontal components from left to right. Normally, for line segmentation for machine printed documents, people use *Projection profile techniques* (Rodrigues et al., 2000). Because the inputs are gray-scale images, so in this stage, they use a modified version of this technique. Let $f(x, y)$ be the intensity value of a pixel (x, y), the vertical projection profile is defined as

$$P(y) = \sum_{x=0}^{W} f(x, y) \tag{3}$$

where W is the width of the image. In the profile, the local maxima corresponds to the gap between lines while the local minima correspond to the text. Because of that, we can also say the main target of this stage is to locate the local maxima.

One problem that might appear is the false local maxima due to noise and to remove that, they smooth the projection function with a Gaussian (low-pass) filter. We can obtain the local maxima by solving for y in:

$$P'(y) = P(y) * G_y = 0 \tag{4}$$

From those local maxima, they can determine the lines and split the image into multiple lines will be used in the next stage.

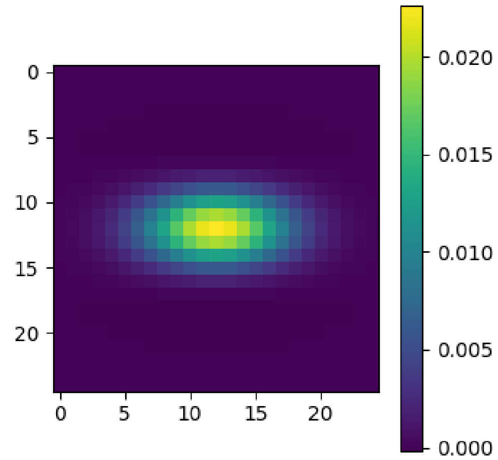
3.3.3 Blob Analysis

The purpose of this stage is to analyze the blob, also, word detection. In the image, a word can be composed of discrete characters, connected characters, or combinations of the two. However, when we focus on words as a unit, the type of character does not affect much, the word is still in a form of a connected region - a blob. The traditional way of forming a blob is to use Laplacian of Gaussian (LoG) (Lindeberg, 1994). Nonetheless, in our work, we used an anisotropic derivative operator.

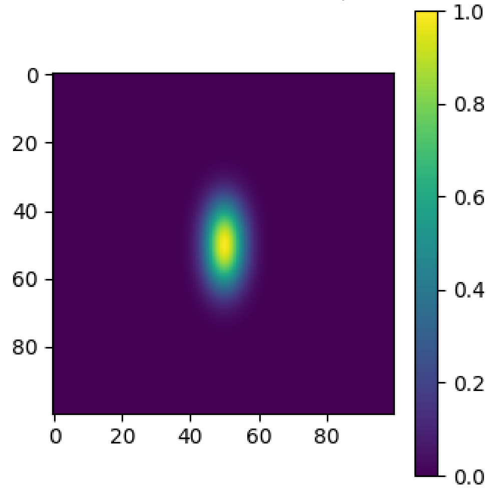
If we observe a word, we may see that the spatial extent of the word is determined by two different factors. The height (y dimension) of the word is determined by the individual characters while the length (x dimension) is determined by the number of character in it. Normally, the length of the word is mostly larger than the height. Due to this reason, the isotropic (same scale in both directions) is not suitable. Instead of that, they use an anisotropic operator (sum of second-order partial Gaussian derivatives along the two orientations at different scales) and choose the x dimension scale to be larger than the y dimension scale so that it will fit with the spatial structure of a word. The anisotropic Gaussian filter is defined as

$$G(x, y; \sigma_x, \sigma_y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-(\frac{x^2}{2\sigma_x^2} + \frac{y^2}{2\sigma_y^2})} \tag{5}$$

The **Figure 4a** shows the filter kernel with size of 25, $\sigma_y = 5$ and **Figure 4b** is its frequency response. It models the typical shape of a word with the width in this case 3 times the height. They also define the multiplication factor $\theta = \frac{\sigma_x}{\sigma_y}$ as the ratio of scales between x dimension and y dimension.



(a) Filter with kernel size is 25, $\sigma_y = 5$, $\theta = 3$.



(b) Frequency Response.

Figure 4: The anisotropic Gaussian filter.

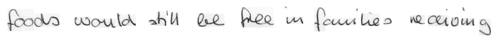
For the above Gaussian, the second order anisotropic Gaussian differential operator $L(x, y; \sigma_x, \sigma_y)$ is define as

$$L(x, y; \sigma_x, \sigma_y) = G_{xx}(x, y; \sigma_x, \sigma_y) + G_{yy}(x, y; \sigma_x, \sigma_y) \quad (6)$$

By convolving the image with $L(x, y; \sigma_x, \sigma_y)$, we obtain the scale space representation of the image. Consider a two dimensional image $f(x, y)$, the output scale space representation is

$$I(x, y; \sigma_x, \sigma_y) = L(x, y; \sigma_x, \sigma_y) * f(x, y) \quad (7)$$

A line image is shown in **Figure 5a**. After applying it with **Formula 7**, the output scale space representation is shown in **Figure 5b**



(a) A line image.

(b) The scale-space representation.

Figure 5: Before and after applying scale-space representation.

3.3.4 Choice of Scale

Although a document image consists of different types of structures such as characters, words, lines at different scales but they do not require different scales to extract different structures. There exists a scale at which each word forms a distinct blob. There is a point we have to notice is that we want to merge character blobs and yet be able to delimit the word. Therefore, they consider a blob as a connected region in space and measure its spatial extent but do not make any volumetric significance for that measurement. The algorithm requires to choose σ_y and the multiplication factor θ. Based on observation, the *maximum of the spatial extent of the blobs* corresponds to the best filter scale. Considering ζ_i as the extent of a blob i, the total extent of blobs for a line is $A = \sum_{i=1}^{n} \zeta_i$.

As we mentioned above, σ_x and σ_y are used to surround the spatial dimensions of a word and $\theta = \frac{\sigma_x}{\sigma_y}$. According to the paper, after doing several analyses, with the constant σ_y, they found out that $\theta \in [3 - 5]$ will give the maximum total of extent of blobs A.

With regard to σ_y, experiments found that σ_y is a function of the height of the words. From the height of a line, we can estimate the value of σ_y as

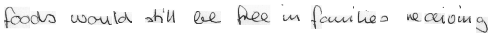
$$\sigma_y = k \times Line\ height \quad (8)$$

where $0 < k < 1$. We can use the same method with finding θ to figure out the best value of k.

The example for the difference in blob images depend on the values of σ_y and θ is shown in **Figure 6**. **Figure 6a** shows the line image and **Figures 6b, 6c, 6d, 6e, 6f** show the corresponding blob images in different cases.

3.3.5 Blob Extraction

In this last stage, the blobs are mapped back to the original image to locate the word. They then created the bounding box using all information archived from earlier stages. However, due to line



(a) A line image.

(b) Blob image with $\sigma_y = 1$ and $\theta = 2$.

(c) Blob image with $\sigma_y = 2$ and $\theta = 2$.

(d) Blob image with $\sigma_y = 4$ and $\theta = 3$.

(e) Blob image with $\sigma_y = 6$ and $\theta = 5$.

(f) Blob image with $\sigma_y = 11$ and $\theta = 7$.

Figure 6: Examples of blob images with different values of σ_y and θ.

segmentation and smoothing, some parts of the words might be lost so the bounding boxes need extending in the vertical direction to capture the full word. There is also a filter used to remove small structures which might appear as the result of noises. The output of this stage in the case of **Figure 5a** is shown in **Figure 7**.

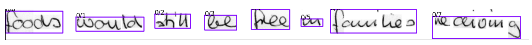


Figure 7: Example for output of Blob Extraction stage.

3.4 Applying on the IAM Handwriting Dataset

For experimental purposes, we use the *IAM Handwriting Dataset*. There are some differences between the work in the paper and the work we conducted. The authors used the historical manuscripts as the inputs while we use the normal English handwriting scripts. The differences between input lead to the requirement of changing the process. Therefore, we have made some improvements to the original algorithm from the paper.

First, the history manuscripts may have been subjected to degradation and introduction of artifacts. Moreover, as a consequence of the photocopying process, there are some lines and margins that need removing. However, the images from the *IAM Handwriting Dataset* have none of those spanners. Furthermore, it has already been in gray-scale mode. Therefore, in our work, the pre-processing stage is unnecessary and is not included. Instead, we only need to crop the header and footer of an image.

Second, the authors did lines segmentation before analyzing and extracting blobs. In our work, we do not the get scale space representation for each of the line images, but the whole image. To analyse the blobs in that scale space representation, we use the *Contour finding method* (Suzuki et al., 1985) implemented in OpenCV (`cv2.findContour` function)[1].

Third, after analysing blobs, in order to segment blobs into lines, we compute the Jaccard distance of y-axis between all pairs of blobs and use the result as the input for *DBSCAN algorithm* (Khan et al., 2014) - a clustering technique. Each obtained cluster corresponds with a line in the image. From that, we can determine which line each blob belongs to.

Figure 8 shows an example of predict result of a sheet from *IAM Handwriting Dataset*.

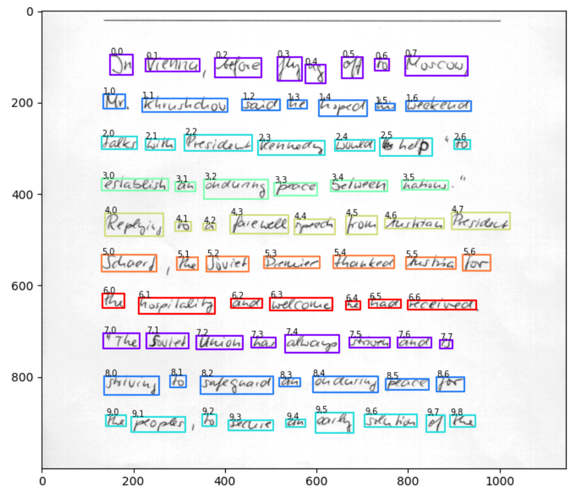


Figure 8: Predict result on the *IAM Handwriting Dataset*.

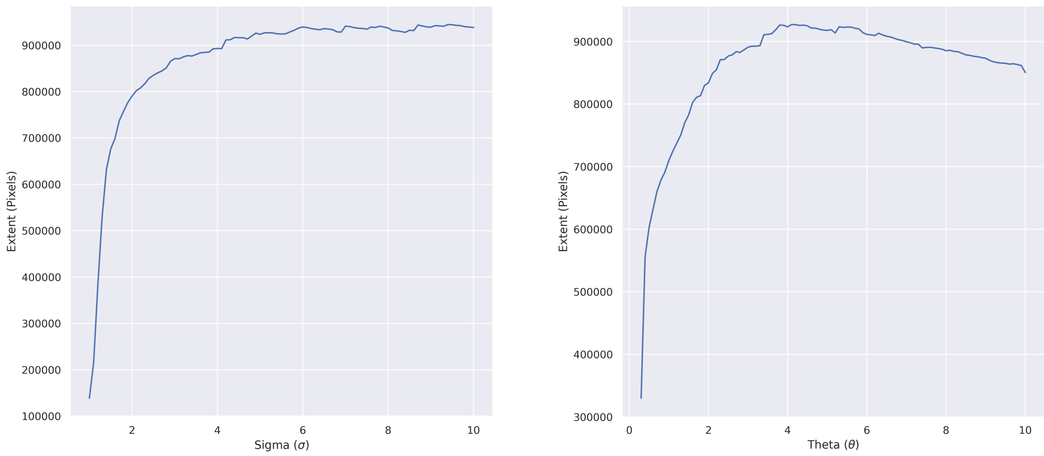
4 Experimentals & Results

4.1 Evaluation metrics

Our target is to evaluate for the objects detection task (in our case is detecting the bounding boxes of words), we use the evaluation metrics called Intersection Over Union (IOU) (Rezatofighi et al., 2019), which is the ratio of the overlap between predict and actual bounding box. For each word and its predicted and actual bounding boxes, the IOU is computed by the following formula:

$$\text{IOU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} \tag{9}$$

[1] `https://docs.opencv.org/4.x/d4/d73/tutorial_py_contours_begin.html`



(a) Between extent and σ, with $\theta = 4$.

(b) Between extent and θ, with $\sigma = 5$.

Figure 9: Correlation between extent and σ, θ.

Then, we can calculate the means of IOU of all words:

$$\text{Means of IOU} = \frac{1}{N} \sum_{i=1}^{N} \text{IOU}_i \tag{10}$$

The range of IOU is between $0 - 1$. The larger the means of IOU is, the more exact our predictions are.

4.2 Parameters tuning

Tuning for kernel size The authors choose the kernel size for the anisotropic Gaussian filter is 25, which is not too large for calculating as well as not too small for deriving word-level scale.

Tuning for σ As we mentioned in **Section 3.3.4**, we can use the total extent of blobs to find the best σ value. With a fix θ value, we can find the best σ value. From **Figure 9a**, we can see that the σ value starts to converge from $sigma > 6$. Hence, we can choose $\sigma \in [5 - 7]$.

Tuning for θ We can use the same method to figure out the best value for θ. From **Figure 9b**, we can see that the θ value reaches the peak at $\theta \approx 4$. Hence, we can choose $\theta \in [3 - 5]$.

Tuning for minimal word area The minimal word area means that we will ignore all the words having the blob's area lower than a pre-defined threshold. After analyzing the area of all words in the dataset, only 3.5% of words with the area lower than 100. So we decided to choose that value as the minimal word area threshold.

Tuning for resize height The resize height is chosen in order to make the average of word height equal to the kernel size (25 – 50 pixels), so that is altered for each image. But based on the characteristic of the *IAM Handwriting Dataset*, the sizes of all handwriting sheets are mostly equal, so we decide to fix the resize height with a value of 800.

4.3 Final results

Because the *IAM Handwriting dataset* is large (5 GB in total) which affects the storage memory and running performance, we decide to use only $1/3$ of that dataset (529 images) for hyper-parameters tuning purposes. We tries multiple combinations of parameters to figure out the best hyper-parameters. **Table 2** shows the list of potential tried values for each parameter:

Parameter	Tried values
Kernel size	25
Sigma (σ)	$5, 6, 7$
Theta (θ)	$3, 4, 5$
Min word area	100
Resize height	700, 800, 1000

Table 2: List of potential tried values for each parameter.

After trying 27 combinations of parameters, **Table 3** shows the top-8 combinations with highest IOU score:

#ID	Sigma (σ)	Theta (θ)	Resize height	IOU
1	6	4	800	0.455
2	5	4	800	0.453
3	5	4	700	0.452
4	7	4	800	0.448
5	6	4	700	0.447
6	5	5	800	0.446
7	6	5	800	0.443
8	5	5	700	0.442

Table 3: Top-8 combinations with highest IOU scores.

For more information about our implementations and evaluations, please visit our Github repository[2].

4.4 Errors analysis

There are 3 main reasons that affect the performance:

1. A word is recognized as two blobs, that leads to the following words even though are detected correctly but are placed in wrong positions.

2. Multi words are recognized as one blob due to the distances between words are close enough to be considered as one component.

3. The sheet contains noises (punctuation, lines, erased components,...).

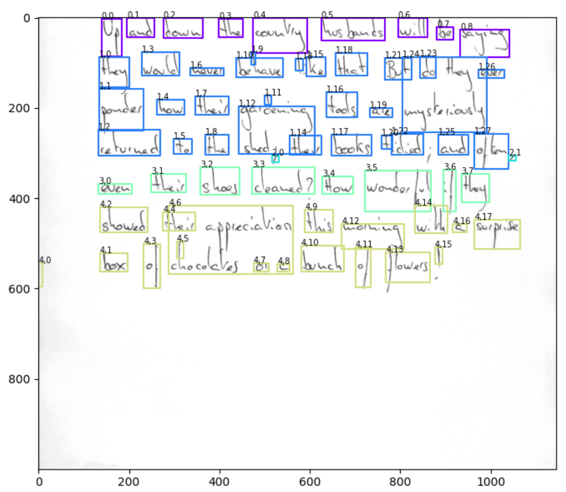


Figure 10: An incorrect detection case.

[2] https://github.com/dqhungdl/WordSegmentation

5 Conclusion

This report presents the work we conducted for our assignment at the Image Processing Course. **(i)** We delivered the content of the technique introduced in the paper, *Word Segmentation with Scale Space Technique* (Manmatha and Rothfeder, 2005), a method to detect words from historical manuscripts. Thanks to the paper, we have learned and got a better understanding not only of this technique but also in the image processing field. **(ii)** Referencing from the research, we re-implemented the method and applied some changes and improvements based on the characteristics of the *IAM Handwriting Dataset*. **(iii)** We also evaluated the results, did some hyper-parameters tuning experiments to produce our best results.

Acknowledgements

Thank you for Manmatha and Rothfeder with the quality research to give us a better understanding in the image processing field, thank you our teacher, PhD. Ta Viet Cuong, and his partners for reviewing our report.

References

Kamran Khan, Saif Ur Rehman, Kamran Aziz, Simon Fong, and Sababady Sarasvady. 2014. Dbscan: Past, present and future. In *The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014)*, pages 232–238. IEEE.

Tony Lindeberg. 1994. Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of applied statistics*, 21(1-2):225–270.

Raghavan Manmatha, Chengfeng Han, and Edward M Riseman. 1996. Word spotting: A new approach to indexing handwriting. In *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 631–637. IEEE.

Raghavan Manmatha and Jamie L Rothfeder. 2005. A scale space approach for automatically segmenting words from historical handwritten documents. *Ieee transactions on pattern analysis and machine intelligence*, 27(8):1212–1225.

U-V Marti and Horst Bunke. 2002. The iam-database: an english sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1):39–46.

Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese.

2019. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 658–666.

Roberto J Rodrigues, Antonio Carlos Gay Thomé, and A Carlos. 2000. Cursive character recognition–a character segmentation method using projection profile-based technique. In *The 4th World Multiconference on Systemics, Cybernetics and Informatics SCI 2000 and The 6th International Conference on Information Systems, Analysis and Synthesis ISAS*.

Nitin P Srimal. 1999. *Indexing handwritten documents*. Ph.D. thesis, University of Massachusetts at Amherst.

Satoshi Suzuki et al. 1985. Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing*, 30(1):32–46.

Tomas Wilkinson, Jonas Lindstrom, and Anders Brun. 2017. Neural ctrl-f: segmentation-free query-by-string word spotting in handwritten manuscript collections. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4433–4442.