ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΑΚΩΝ & ΕΠΙΚΟΙΝΩΝΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Μεταπτυχιακό Πρόγραμμα Σπουδών

«Διαχείριση Πληροφορίας και Τεχνολογίες Παγκοσμίου Ιστού»

<u>ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ</u>

ΤΙΤΛΟΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

“A Semi-Automatic System for Fine-tuning Binarization Algorithms”

Ονοματεπώνυμο φοιτητή:

Βέρρας Βασίλειος

Ονοματεπώνυμο Επιβλέπουσας Καθηγήτριας:

Καβαλλιεράτου Εργίνα

ΚΑΡΛΟΒΑΣΙ

ΦΕΒΡΟΥΑΡΙΟΣ, 2013

ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΙΓΑΙΟΥ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΑΚΩΝ & ΕΠΙΚΟΙΝΩΝΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Μεταπτυχιακό Πρόγραμμα Σπουδών

Η Διπλωματική Εργασία

παρουσιάστηκε ενώπιον

του Διδακτικού Προσωπικού του

Πανεπιστημίου Αιγαίου

“A Semi-Automatic System for Fine-tuning Binarization Algorithms”

Βασίλειος Παναγιώτη Βέρρας

Summary: In image processing a basic step in order to extract useful information is the binarization of the image. Most binarization algorithms take a number of parameters that affect the output of the process. Finding the best settings of a binarization algorithm is time consuming and requires a lot of effort especially when an algorithm depends on many parameters thus the search space is big. To address this issue it is essential to develop a system that finds these parameters that are as close as possible to the best parameters for a given case within an acceptable time frame without having to run through the entire search space.

This thesis aims at developing such a system that allows the user to find the closest to the best settings of a binarization algorithm either automatically using simulated annealing or interactively using a user feedback framework that enables the user to evaluate the results of image binarization. This system incorporates three sample algorithms but it is also expandable allowing for the integration of user developed algorithms.

Keywords: image binarization, simulated annealing, parameters, automatic, feedback.

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω την επιβλέπουσα καθηγήτρια αυτής της διπλωματικής εργασίας κ. Καβαλλιεράτου Εργίνα για την υπομονή, την στήριξη, την καθοδήγηση και την βοήθεια που μου παρείχε καθόλη την διάρκεια εκπόνησης της. Θα ήθελα επίσης να ευχαριστήσω τη σύζυγό μου Ράνια για την ακούραστη υποστήριξη της όλο αυτό τον καιρό χωρίς την οποία δεν θα ήμουν σε θέση να φέρω σε πέρας αυτή την εργασία.

Table of Contents

List of Figures

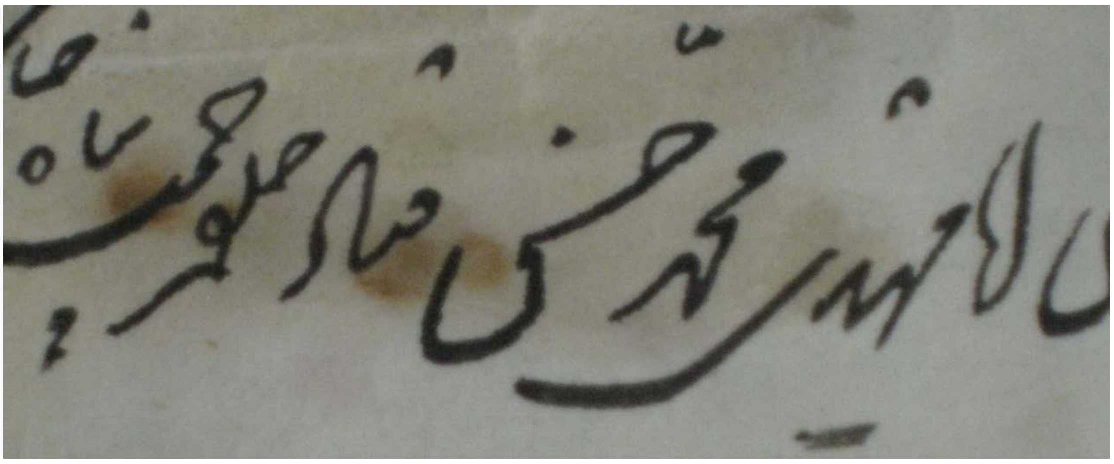
List of Tables

1. Introduction

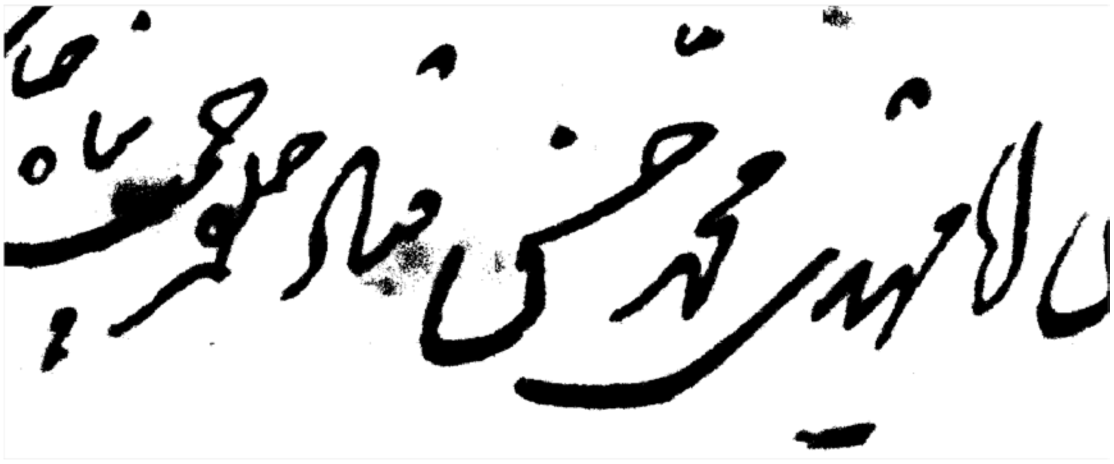
1.1 Image Binarization Problem

Document image binarization primarily concerns the processing of historical documents that contain important information about a person, place or event in the past. The preservation and publishing of historical documents is of utmost importance. Due to the fact that the storage of digital versions of the documents needs huge storage space in order to preserve the properties of the original image, it is not easy to distribute it through the internet. One solution is to convert the image into a bi-level or binary image by finding and applying a threshold on the pixels of the image. The pixels then are classified into two classes; foreground and background. Foreground refers to pixels that belong to useful information such as text, images and tables and correspond to the ink of the original document and is represented in black. Background refers to the paper or other material of the original image and is represented in white. Image binarization is the initial step of most document image analysis and processing in order to subsequently obtain useful information with other methods such as Optical Character Recognition (OCR) or simply convert the document to a more appealing form without problems caused by several degradation issues such as smear, strain, non-uniform illumination, shadows, bleed-through (when the ink transposes from one side of the paper to the other side) etc. The better the document is processed during this phase, the easier it is to process it in subsequent phases and the more useful information can be automatically extracted.

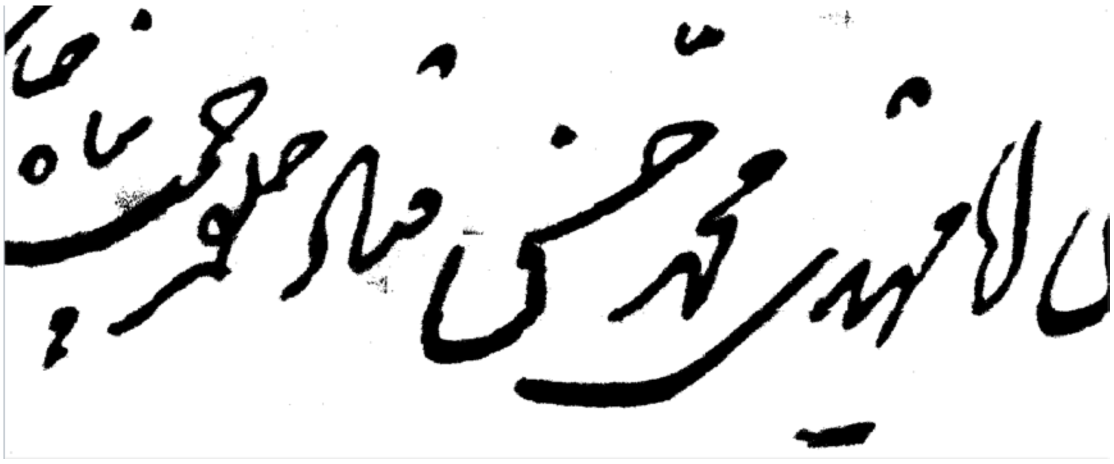
There are two approaches as far as document binarization is concerned; local and global. The easiest approach is to apply a global threshold on the document and decide upon this threshold. While this approach is good and adequate for documents with good illumination and contrast, the aforementioned degradation issues can severely affect its performance. In this case the threshold needs to be decided and applied locally. In order to achieve this, the document is segmented using various methods like sliding or fixed windows or by roughly estimating the foreground and the background areas. Then the threshold is calculated within these areas based on local features. Moreover, there are approaches like [1] that combine global and local methods to achieve a better performance. The effect of a local binarization versus a global approach is shown in Figure 1.



(a)



(b)



(c)

Figure 1. (a) Original Image, (b) Global binarization and (c) Local with W=140

For this demonstration a new algorithm described later in this thesis was used with a 140X140 pixel window partitioning and a threshold of 85% of the mean intensity used globally (case b) or locally in each window (case c). It is clear that local binarization helped distinguishing between text and noise around the text and even on the top right part of the image.

Most algorithms have one or more parameters that control the way the document is processed and may heavily affect the binarization result. These parameters usually control the global or local threshold, the size of the window used to scan the document, the amount of the standard deviation of pixel intensity added to the average intensity and many others. Typical examples are Niblack’s [2] algorithm that uses a sliding window to apply a local threshold based on the mean and standard deviation of the pixels’ intensity in the window and [1] that iteratively processes the image globally and then based on fixed window segmentation decides whether more local processing is required.

The number of parameters as well as the range of values for each parameter defines the search space of the algorithm. This search space can be enormous and consequently the time and effort required to search through it could be a limited factor towards good binarization performance. Due to the fact that different parameters are required for different set of images the performance of an algorithm depends not only on its sophisticated design but also on the user’s ability to fine tune it in a reasonable time frame.

1.2 Scope of this thesis

All things considered, this thesis aims at developing a system that drastically reduces the time and effort needed to find the appropriate configuration of a binarization algorithm that produce if not the best, very close to the best binarization results for that specific algorithm and collection of documents. In general, the system incorporates a probabilistic method called Simulated Annealing that can be used to explore the configuration settings of an algorithm that behave well when applied to documents categorized by degradation issues such as smear, stain or low contrast or find the configuration settings that work best in most cases. Taking all the above into account, it is obvious that the system is ideal for fine tuning algorithms when processing large collections of historical documents where all documents contain more or less the same degradation issues. This is true for collections of documents were all documents were stored together and digitized using the same method (scanner of camera). These documents suffer uniformly from the same type and amount of degradation issues. In this case, the algorithm can be fine tuned using manually created Ground Truth (GT) documents based on a small number of representative documents from the collection. An estimation of the number of documents required is given in §6.3. The GT documents can be created based on the user’s personal preference on the quality of the resulting image. The user for example could opt for a document with thicker strokes. The algorithm then is tuned to adapt to the desired GT document and thus the whole collection is processed in the same manner.

Another use of the system could be to explore the default settings of an algorithm that is the settings that are more likely to work well under all circumstances. This can be achieved by creating a collection of images with their respective GT images that suffer from all possible degradation issues. Then the settings of the algorithm can be found that produce the best overall performance. These settings can then be used as the default setting of that algorithm.

2. Related work

Automatic or semi-automatic tuning of binarization algorithms has been given limited attention to date. Badekas and Papamarkos [3] propose a Parameter Estimation Algorithm (PEA), which can be used to detect the best values for the parameter set of every binarization algorithm. They use edge detection evaluation to analyze the correspondence between different binarization results obtained by the application of an algorithm to a specific document using different parameters. They improve on previous defined techniques [4] by applying an adaptive convergence procedure to reduce the parameter’s ranges by estimating the best and second best binarization result. To evaluate the results of each binarization, they estimate a ground truth image (EGT) selecting it from a list of Potential Ground Truth (PGT) images obtained from a technique proposed by Yitzhaky and Peli [4]. These PGT images are produced using N images derived from different binarization techniques. Every PGT_i ($0 \le i \le N$) is created by taking all the pixels that are classified as foreground in at least i images. The rest is background. Then the best PGT image is designated as EGT using ROC analysis and Chi-square test.

Mohamed Cheriet, Reza Farrahi Moghaddam and Rachid Hedjam [5] introduce a framework for the optimization of parametric binarization methods, which provides the optimal values for each document image. Numerical feature vectors for two-dimensional data are generated based on their maps obtained by the use of Stroke Gray Level (SGL) multilevel classifier. They then combine the statistics of various maps in a nonlinear way to produce the final feature vector. Finally, the optimal behavior is learned using Support Vector Regression (SVR). To evaluate the framework they used the grid based Sauvola method and Lu’s method against the DIBCO’09 and H- DIBCO’10 datasets for comparison.

Nicholas R. Howe [6], apart from a new binarization technique, introduces an automatic parameter tuning by means of a stability heuristic criterion that helps to choose suitable parameter values for individual images. The technique is based on the observation that when good parameter values are found, small changes to them will give low variability in the binarization result. However, due to the computational cost of the proposed solution, the author applies the proposed stability criterion only to one of the two parameters of the suggested algorithm, restricting the other to two possible values. When minimizing the first parameter, small changes to it will result to the same behavior as far as minimizing is concerned. This can lead to a substantial reduction in the computational cost. Moreover, two distinct values of the second parameter, at the edges of its range, are enough to produce most of the data required for the optimization. Although the author’s approach works well on the proposed algorithm, it heavily depends on the specific behavior and parameters of it, making it almost impossible to generalize the approach to other binarization techniques. On the positive side, this approach does not depend on nor computes a ground truth image.

3. The three included algorithms

The system includes three algorithms; one hybrid global-local thresholding and two local thresholding. *Fixed Window Local Thresholding (FWLT)* is a new technique created to demonstrate that even when a simple and naïve approach is used, if tuned properly produces satisfactory and comparable binarization results. The other two are Niblack’s algorithm [2] and a hybrid binarization technique [1] that combines iterative global thresholding with local post-processing.

3.1 Niblack

Niblack’s algorithm uses a sliding square window over the image, centered on a pixel, and calculates a threshold T for that pixel using the equation:

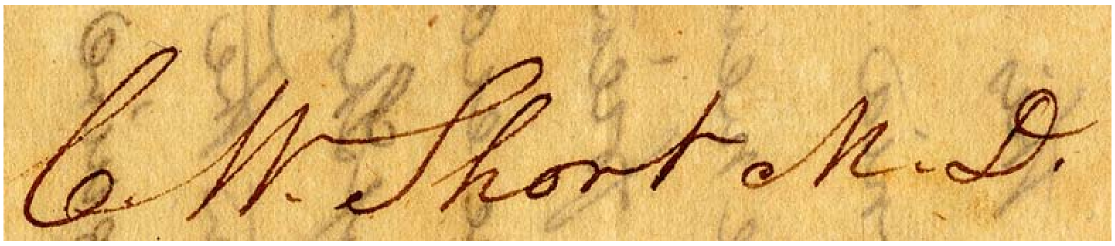
$$T = m + k * s$$

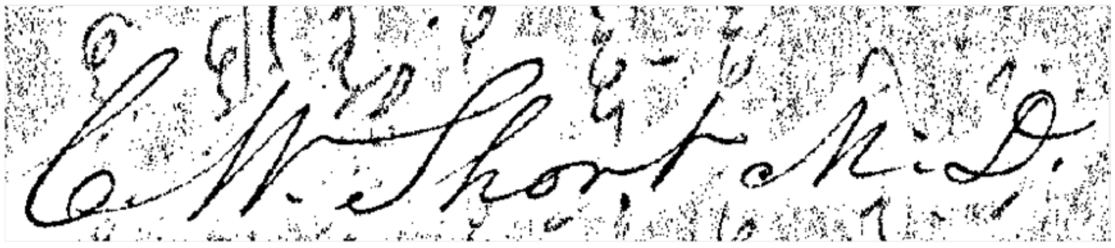
where m is the mean intensity value of all the pixels in the window and s is the standard deviation. *k* is a configuration value that along with the window property *h* affects the quality of the produced image. *s* is denoted as:

$$s = \sqrt{\left(\frac{1}{h*h-1}\right) * \sum_{i=0}^{h*h} (intensity(i) - m)^2}$$

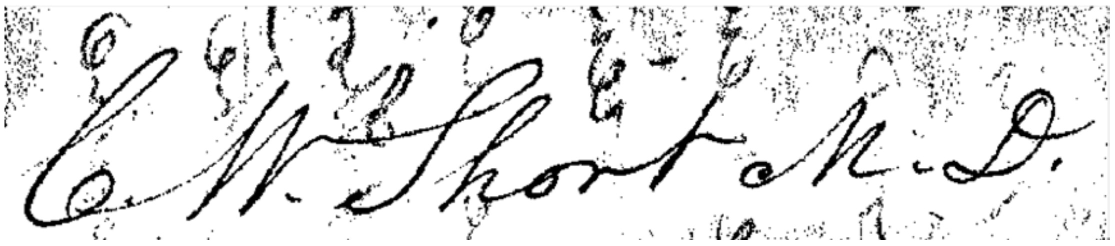
The window used is calculated around the pixel with a side of $2 * h + 1$. It does not use any pre or post-processing on the document. The system uses as initial values for *k* and *h,* 0.6 and 25 pixels respectively as described in [7].

The algorithm’s Achilles' heel is the fact that it cannot cope with large areas of background pixels. This is evident in Figure 2.

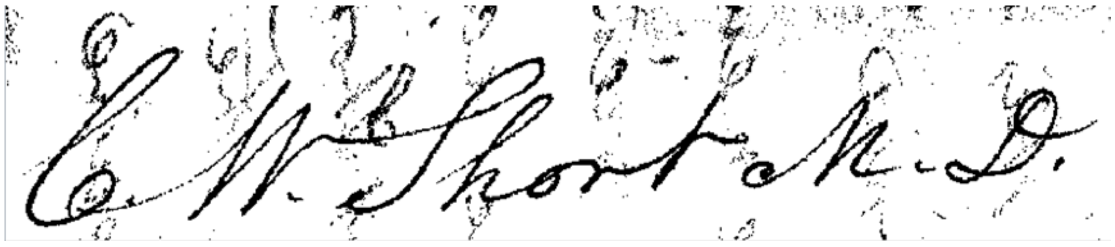

(a)



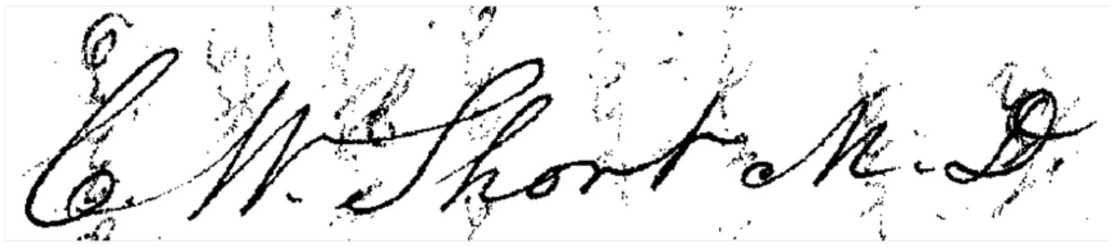
(b)



(c)



(d)



(e)

Figure 2. Niblack Binarization (a) Original Image, (b) k= -1.0 h=15, (c) k=-1.0 h=30 (d) k=-1.0 h=50 (e) k=-1.0 h=100

When a pixel is outside a distance of 15, 30, 50 or 100 pixels respectively from foreground pixels (text) then the pixel is more likely to be misclassified as the mean and standard deviation of the window is calculated using only background pixels. It is obvious that the bigger the window is, the more likely it is for a pixel to be within window distance to real foreground pixels and consequently the more likely it is to be correctly classified. However, the bigger the window it is the slower the algorithm becomes. This problem led [7] to introduce an adaptive Niblack that determines the areas where *k* and *h* can be applied and background areas where all pixels are classified accordingly.

3.2 Hybrid Iterative Global Thresholding (H-IGT)

This approach is done in three steps:

- Application of Iterative Global Thresholding Algorithm
- Detection of “Noisy” Areas, and
- Application of Iterative Global Thresholding Algorithm to the detected areas.

- Application of Iterative Global Thresholding

This method performs a number of iterations on the image until one of two conditions is met. At the beginning of each iteration i the global average pixel intensity T_i is computed. Then the following steps follow:

- Subtraction of T_i from each pixel.
- The grayscale histogram is stretched so that the remaining pixels are distributed in all the grey scale tones.

When either $|T_i - T_{i-1}| < termination\ criterion\ (tc)$ or a given number of iterations i is met the procedure stops and the resulting image is produced by turning all non white (1) pixels to black (0). Normally, tc will occur before i is met. Then, tc controls the number of iterations over the image. Lower values of tc results in more iterations applied and hence more pixels classified as background resulting in thinner strokes with less noise. The formula used for the subtraction that provides the after-subtraction and before-equalization image I_s is:

$$I_s(x,y) = I_i(x,y) - T_i + 1$$

The relation used for the histogram stretching is:

$$I_{i+1}(x,y) = 1 - \frac{1 - I_s(x,y)}{1 - E_i}$$

where I_s is given by the previous equation and E_i is the minimum pixel value in the image I_s during the i^{th} repetition, just before the histogram stretching. After every repetition, more pixels move to the background thus creating a cleaner image. As suggested in [1], an upper limit of 20 is posed for i.

- Noisy area detection

At this stage the image is divided into segments (S), of fixed size $n * n$. In each segment, the frequency of black pixels is calculated. The segments that satisfy the following criterion are, then, selected as:

$$f(S) > m + k * s$$

where $f(S)$ is the frequency of the black pixels in the segment S while m and s are the mean and the standard deviation of the black pixel frequency of the entire page, respectively. The parameter k in the formula determines the sensitivity of the detection method. The higher the k, the fewer segments will be detected.

- Re-application of IGT (Local Thresholding)

The IGT method is then applied to the selected segments of the document. The iterative procedure stops when either $|T_i - T_{i-1}| < tc$ is satisfied or the number of iterations required in the initial global thresholding is exceeded.

It is obvious that the performance of the H-IGT algorithm depends on the value of the parameter k and the size n of the window used in local thresholding as well as on the termination criterion tc. Despite what is suggested in [1], the system uses as initial values for k, n and tc, 2.5, 50 pixels and 0.4 respectively. These settings were provided by the author based on new unpublished experiments on the algorithm.

3.3 Fixed Window Local Thresholding (FWLT)

This algorithm is a simple and fast local binarization method that like H-IGT uses a fixed square window to segment the image and decides on the threshold based on a given percentage of the mean intensity value of all the pixels in the window:

$$T = k * m$$

where *T* is the calculated threshold and *k* is the given percentage. Apart from *k,* the quality of the produced image depends on the window parameter *w* as in previous algorithms. The system uses as initial values for *k* and *w,* 82 and 50 pixels respectively.

Despite the simplicity of the algorithm it has certain advantages especially when compared to Niblack. Using a fixed window makes the algorithm faster. In fact the bigger the window is, the faster the algorithm performs. On the other hand a fixed window means that the algorithm cannot adapt to each pixel individually as in the case of a sliding widow where the pixel to be classified lies at the center of it. But as was demonstrated in Figure 2 the best results were achieved in bigger windows resulting in very slow performance from the algorithm. The application of the algorithm in the same document as in Figure 2 is show in Figure 3.

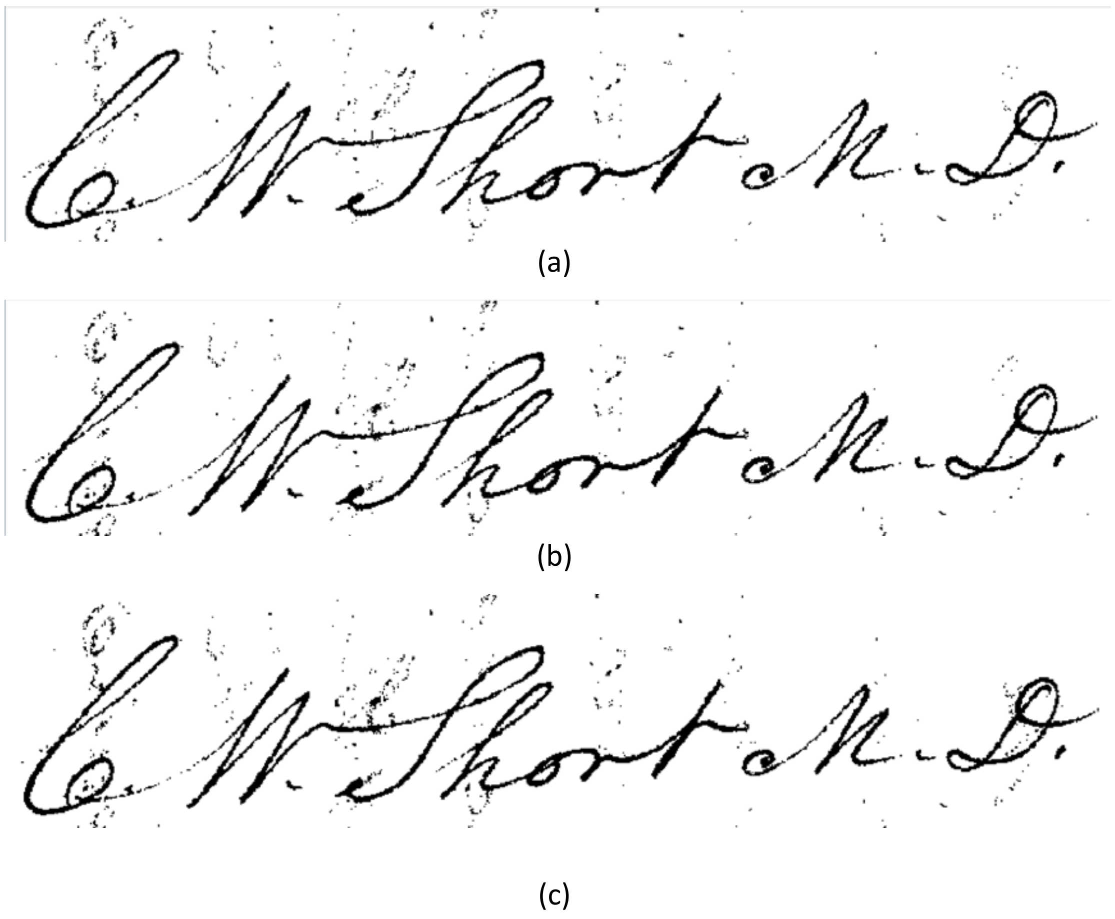
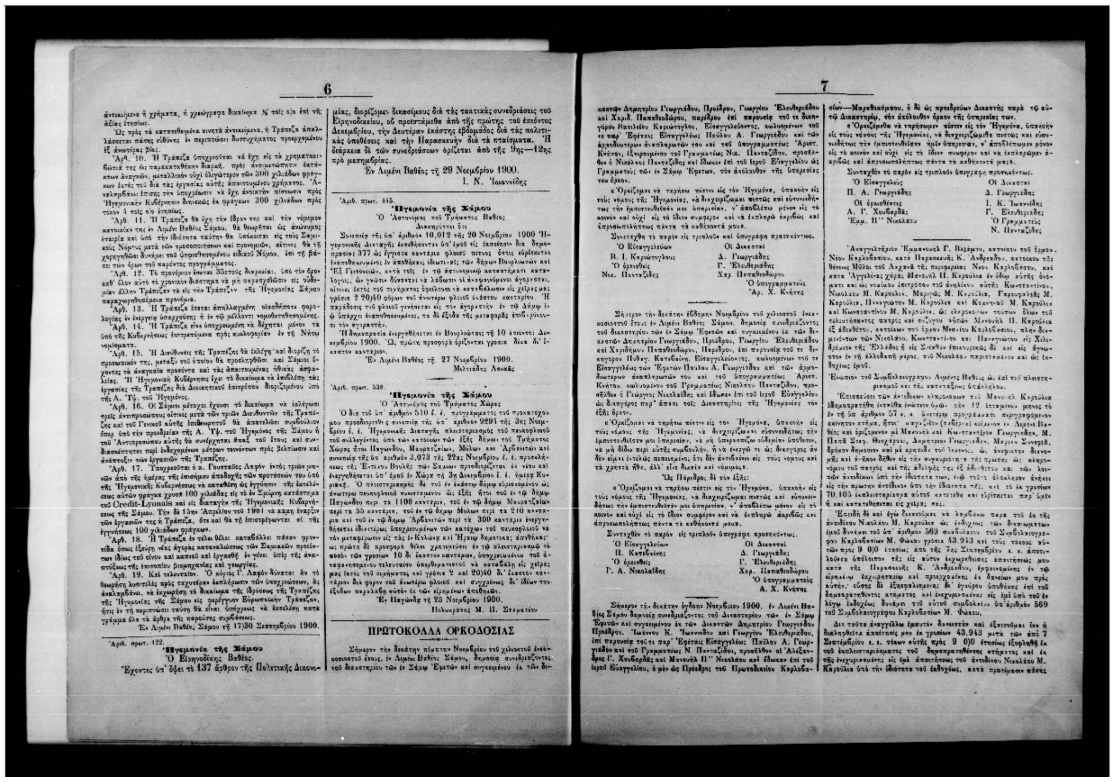
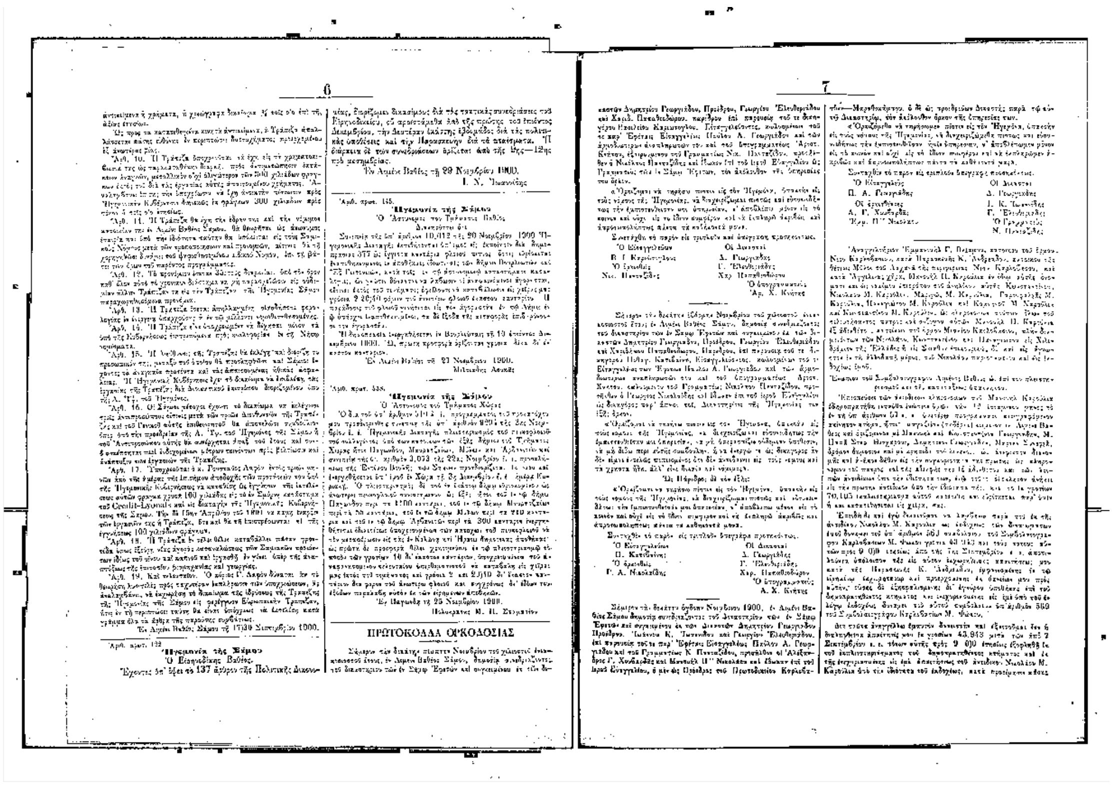


Figure 3. FWLT (a) k=70 w=25, (b) k=70 w=50, (c) k=70 w =100

Another advantage of FWLT is that it can correctly classify big background areas even if they consist of pure black pixels. This is demonstrated in Figure 4.


[illegible]

(a)


[illegible]

(b)

Figure 4. FWLT on Samos Historical Archive Document (a) Original, (b) k=88 w =20

As long as k is bigger than the biggest text stroke in the document the algorithm will correctly classify big black areas as background without harming useful text foreground pixels.

4. Simulated Annealing

Simulated Annealing (SA) is a metaheuristic probabilistic optimization method used to exploit the minimum (or maximum) of a given function that depends on many parameters (combinatorial optimization). Osman and Laporte in [8] describe metaheuristic methods as follows:

“A metaheuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solutions.”

Simulated Annealing was first introduced by Kirkpatrich, Gelatt and Vecchi [9] who developed the method based on previous work from Metropolis et al. [10]. Metropolis et al. described the behavior of molecules during the cooling phase of liquids or metals in metallurgy (annealing phase) from moving freely at high temperatures until they find equilibrium at a lower temperature. Kirkpatrich, Gelatt and Vecchi introduced a method that simulates this behavior.

Most minimization strategies find the nearest local minima by simply accepting every solution that improves the best solution found so far. Simulated Annealing avoids local minima by probabilistically accepting solutions that do not improve the best solution so far. The process begins from an initial state with some random or default parameters. Then these parameters are given a small random displacement. The overall difference of energy (ΔE) from the previous step is then computed. Our goal is to find the lowest possible energy state so if $\Delta E <= 0$ the new state is accepted and a new iteration begins based on the new energy state. If not then the new state is probabilistically accepted or rejected based on a random number generated between 0 and 1 and the current temperature. This allows for parameter space exploration at high temperatures. This exploration is restricted as the temperature drops. This enables the system to go to higher energy states avoiding local minima that would otherwise trap the process. The slower the process proceeds, the higher the probability that this final energy state is the lowest between all the energy states that the system can have.

It is straightforward that this method can be used in document binarization since it is a combinational optimization problem as almost all algorithms depend on one or more independent parameters. The energy of the system can be computed based on an evaluation measure of the binarization result at each step. Our system uses an implementation of SA taken from Prokopiou K. [11] published as an open source project (available at https://github.com/kprokopiou/RuLieR) who implemented it in a system that optimizes page rule-line removal algorithms. This implementation was modified and optimized to work in our system.

4.1 Detailed implementation of Simulated Annealing

The process starts at an initial state where the default algorithm parameters are used unless overridden by the user. The system allows to start from every parameter settings allowed by the algorithm. For the energy calculation the system compares the produced by the algorithm binarized image to a Ground Truth (GT) image. For the comparison a widely used evaluation measure (F-Measure or Harmonic Mean) is used. F-Measure is defined as:

$$F - Measure = \frac{2 * Recall * Precision}{Recall + Precision}$$

where:

$$Recall = \frac{TP}{TP + FN}$$

and

$$Precision = \frac{TP}{TP + FP}$$

TP stands for True Positive is the total number of foreground (black) pixels in the binarized image that are also foreground in the GT image.

FN stands for False Negative is the total number of background (white) pixels in the binarized image that are foreground in the GT image.

FP stands for False Positive is the total number of foreground (black) pixels in the binarized image that are background in the GT image.

Precision can be seen as a measure of quality as it answers the question “How many of the foreground pixels identified are actually foreground?” and recall a measure of quantity as it answers the question “How many of the total foreground pixels were correctly identified?”. The F-Measure combines precision and recall in an even way that gives a better representation of the similarity between the GT image and the produced binarized image.

Our goal is to maximize F-Measure meaning that higher F-Measure leads to lower energy states. Maximum value for F-Measure is 1 or 100% when all pixels are correctly predicted thus the binarized image is identical to the GT image.

As in [11] the calculation of the temperature is based on Press et al. [12] and is given by the following formula:

$$T = T_0 * [1 - \left({}^{t}/_{MAX\_ITERS} \right)]^a$$

where T_0 is the initial temperature, t is the current iteration, MAX_ITERS is the maximum number of iterations allowed and a is a constant number that can be usually set to 1, 2 or 4 depending on the position of the relative minima. Large values of a will spend more time at lower temperatures. The system uses $T_0 = 100$, $MAX\_ITERS = 45$ and $a = 1$.

The system calculates the energy of the initial state using the default settings as described above. Then a new parameter set is calculated by randomly setting one parameter taken from a small neighborhood around the old value. The neighborhood is calculated as follows:

- First the half range of the neighborhood is calculated using the following equation:

$$halfRange = \frac{(initRange * k^3)}{3}$$

where k is a loop counter starting from 1 and increased by 1 every time a new solution is rejected and $initRange$ is the initial range for that specific parameter. In that way the neighborhood grows bigger in case the new energy state is not accepted allowing for wider spectrum exploration.

- Then the lower and upper bound of the neighborhood is calculated:

$$lowerBound = currentValue - halfRange$$
$$upperBound = currentValue + halfRange$$

- A check is performed to align the new borders to the initial range of the parameter in case they exceed it. Finally, the new value is calculated as follows:

$$newValue = lowerBound + randomDouble * (upperBound - lowerBound)$$

Using the new parameter set, a new binarized document is produced and its energy is calculated using F-Measure as described. The difference between the old and the new energy $\varDelta E$ is then calculated. If $\varDelta E \leq 0$ then the new state is accepted and a new iteration begins. If $\varDelta E > 0$ then the new state is not discarded but a probability P is calculated based on the following formula:

$$P = \left(\frac{T}{T_0}\right) * e^{\frac{-\varDelta E}{T}}$$

where T is the current temperature and T_0 is the initial temperature. If P is bigger than a random number generated between 0 and 1 then the new state is accepted otherwise the old state is retained and a new iteration begins. When either the maximum number of iterations MAX_ITER is reached or the lowest possible energy is achieved the process terminates.. In Figure 5 below the flowchart of the process is depicted.

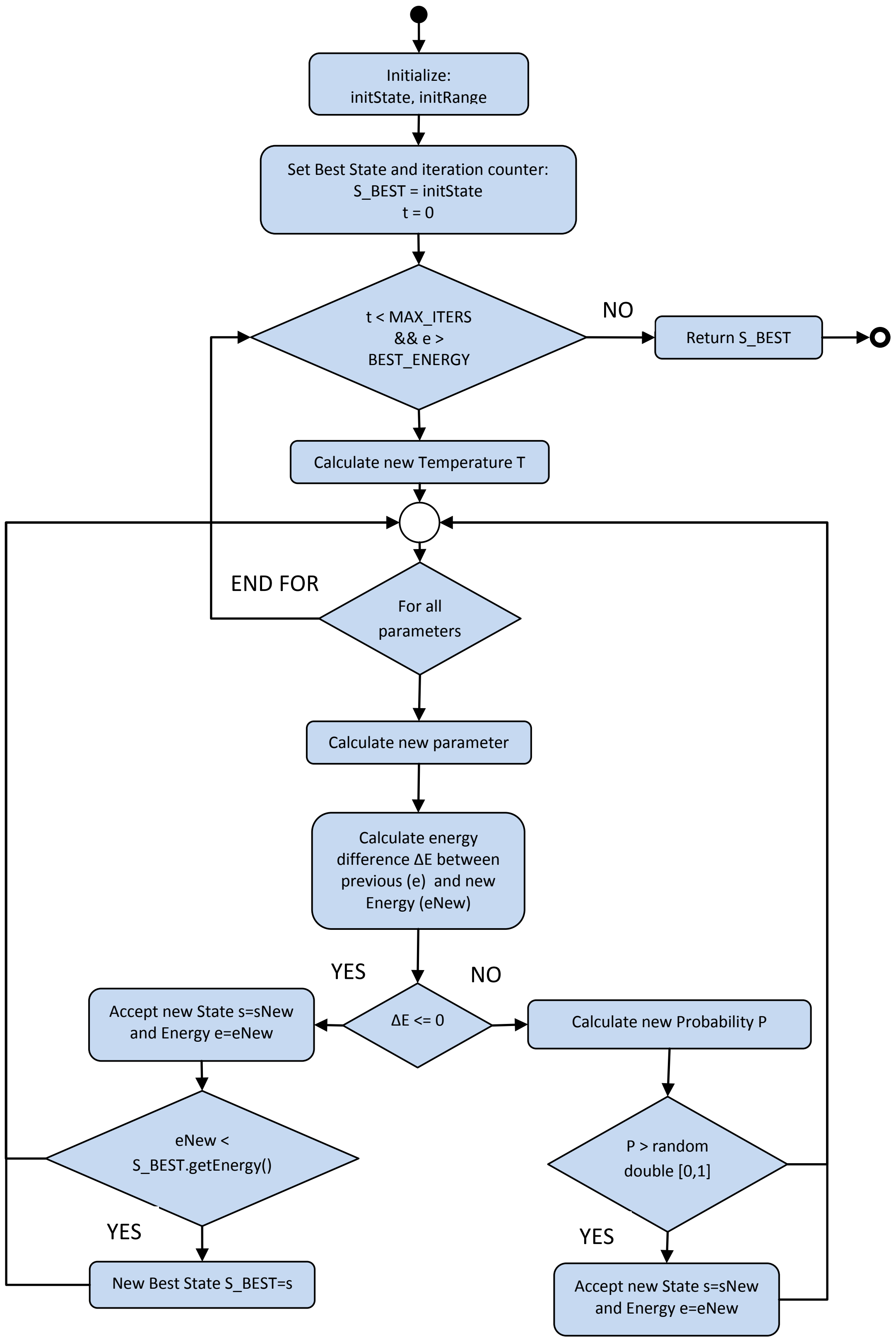


Figure 5. Simulated Annealing flowchart

5. Implementation of the system

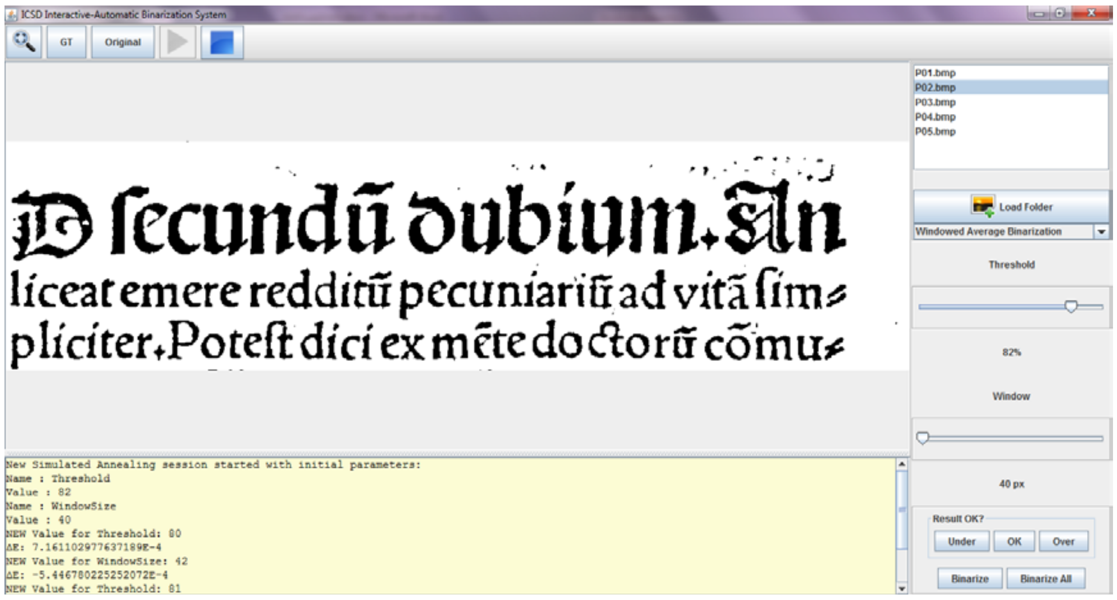
5.1 Tools used

The programming language used to implement the system is Java^{TM} [13] in particular Java^{TM} Standard Edition Development Kit (JDK^{TM}) version 7 update 7 [14] and later versions. Java was chosen due to the fact that it is popular, easy to use, object oriented, expendable and platform-independent, making it the best candidate for research purposes without any limitations and cost. It allows you to create modular programs and reusable code. Moreover, there is a number of free existing libraries written in Java for image I/O, display, processing etc saving the developer from important development time. Most of the image handling is done by the Java 2D API which is incorporated into the JDK^{TM}. However, in order to support more image formats such as TIFF, the Java Advanced Imaging v1.1.2 [15] is used.

For the development of the system Netbeans^{TM} 7.1.1 Integrated Development Environment is used freely available at [16].

5.2 Main Window

The Extendable Image Binarization Tool’s main window (fig.1) consists of four areas; the toolbar, the image display area, the algorithm selection, configuration and application area and the log area.



```
New Simulated Annealing session started with initial parameters:
Name : Threshold
Value : 82
Name : WindowSize
Value : 40
NEW Value for Threshold: 80
ΔE: 7.161102977637189E-4
NEW Value for WindowSize: 42
ΔE: -5.446780225252072E-4
NEW Value for Threshold: 81
```

Figure 6. Main Window

5.2.1 Toolbar

The toolbar area contains the Fit-To-Window button that fits the image to the image display area and the Simulated Annealing panel.

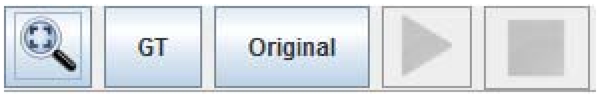


Figure 7. The Toolbar

The Simulated Annealing panel is responsible to load the Ground Truth (GT) and Original image folders and start a new session. In order for the "start" button to be activated, these folders must contain the same number of image files. There are no naming conventions for the images in the folders. However, the alphabetical order of the returned image lists must hold the GT images of the corresponding Original images at the same index. Once the "start" button is activated a new simulated annealing session can be started. The initial configuration settings are taken from the current settings of the selected algorithm. During the session the "start" button is deactivated and the "stop" button is activated, enabling the user to terminate the session at any time before it is completed. During the session useful information is displayed on the Log Area.

5.2.2 Image Display

The image display area displays the selected image or the produced image after a "Binarize" action is performed. The user can zoom in/out using the mouse scroll button and pan by left click, hold and drag. Note that every time a "Binarize" action is performed the binarization is done on the original image and the resulting image is displayed.

5.2.3 Algorithm Selection, Configuration and Application

This area contains from top to bottom the following subareas:

- Image folder selection.

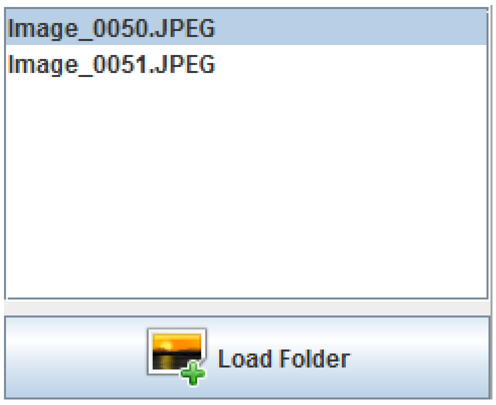
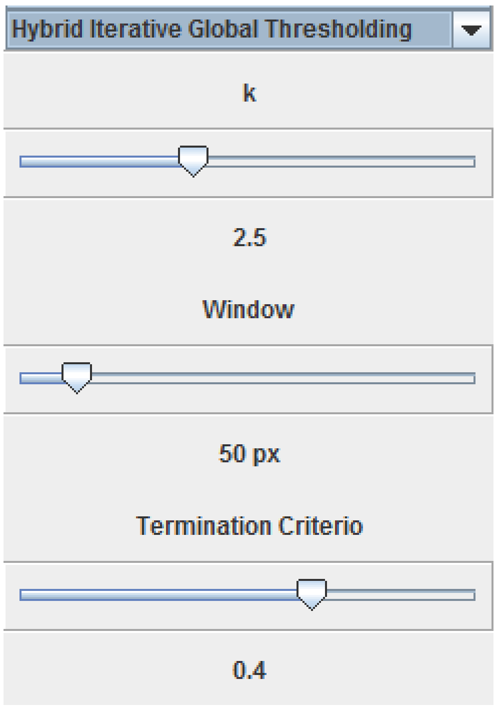


Figure 8. Image folder selection

In this subarea the user can designate a folder that contains the images for binarization. This can be done by left clicking on the “Load Folder” button and browsing through the desired folder from the popped-up file selection dialog. Once the folder is selected, the list above the button is populated with all the images the folder contains. The user can then select individual images from the list by left clicking on the image name. The image then is displayed on the image display area.

- Algorithm selection and configuration.


Hybrid Iterative Global Thresholding

k

2.5

Window

50 px

Termination Criterio

0.4

Figure 9. Algorithm selection and configuration

In this subarea the user can select the desired binarization algorithm from the drop-down list. Once the algorithm is selected the appropriate algorithm configuration is displayed with the specific algorithm settings.

- User feedback and algorithm application.

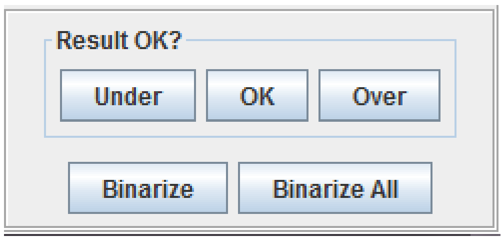
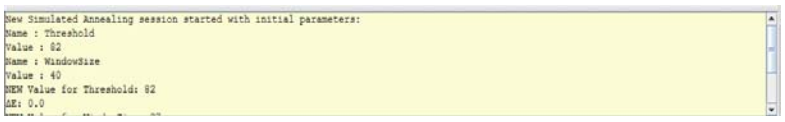


Figure 10. User feedback and algorithm application

In this subarea the user can apply a binarization on the selected image based on the current settings and provide a feedback on the results of the binarization. The user has three options; under, ok and over. Under should be selected when the result of the binarization contains noise more than desired. Over should be selected when the binarization produces a “white” image with part of the valuable information missing. Ok should be selected when the result is satisfactory. Based on the implementation of each algorithm, selecting one of the feedback buttons can result in the automatic reconfiguration of the algorithm towards new settings that should produce images closer to the desired level of binarization. The user, however, can override this proposal and fine tune the algorithm manually. Once the correct settings are determined the user can apply the algorithm to all the images in the selected folder by selecting the “Binarize All” button. The images produced are stored in the same folder with the same name extended with an abbreviation of the current name and settings of the algorithm used. Log files with the results of each step are produced for further processing. The log file framework is explained in the following section.

5.2.4 Log Area



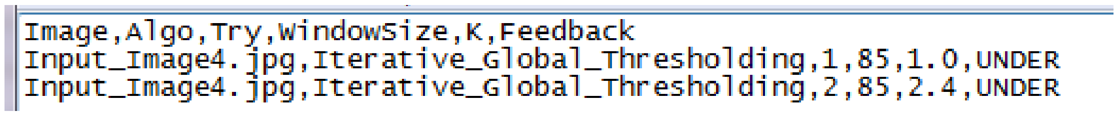
```
New Simulated Annealing session started with initial parameters:
Name : Threshold
Value : 82
Name : WindowSize
Value : 40
NEW Value for Threshold: 82
ΔE: 0.0
```

Figure 11. Log Area

The Log text area is used to display useful information regarding the process of each Simulated Annealing process. In particular the current algorithm settings as well as the best PERR and the Difference of Energy (ΔE) of each binarization are displayed. At the end of the session a summary of the best settings is displayed.

5.3 Logging

The first time the application starts, a folder called “Log” is created in the same path the application was started from. This folder contains all the log files produced by the application. Each time a new algorithm is selected a new log file is created with the name of the algorithm and the current date and time. The log file (fig.2) is a comma separated value (csv) file with a header containing the image name, the algorithm used, the number of tries, the names of the different settings and the feedback, and one line for every user feedback input with the corresponding values.



```
Image,Algo,Try,WindowSize,K,Feedback
Input_Image4.jpg,Iterative_Global_Thresholding,1,85,1.0,UNDER
Input_Image4.jpg,Iterative_Global_Thresholding,2,85,2.4,UNDER
```

Figure 12. Log file contents

6. Evaluation of the system

The evaluation of the system is done based on four pylons. The capabilities of the system are shown applying the Simulated Annealing (SA) process to the DIBCO 2011 dataset using the Hybrid-Iterative Global Thresholding algorithm. The goal is to show that the system can improve the binarization results in comparison to the default settings of the algorithm. Then the ability of the system to consistently find very close to the best configuration settings is demonstrated by finding FWLT's best settings for a single document using Brute Force and then comparing the binarization results and the total time taken to the results obtained using multiple SA sessions. Moreover, an attempt is made to estimate the number of GT images required for the SA process to provide settings that can be applied to the entire dataset with the best results. Finally, a comparison is done between all methodologies and algorithms used in this system.

The configuration of the system that the evaluation was performed on is shown below:

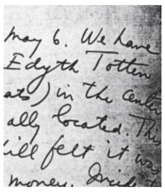
System Type	Notebook
Processor	AMD C-60 APU with Radeon^{tm} HD Graphics 1.00 GHz
Installed Memory (RAM)	4.00 GB
Operating System	Windows 7 Home Premium SP1 64-bit
Java Runtime Environment (JRE)	Java^{tm} SE RE build 1.7.0-b147

Table 1. Evaluation System Configuration

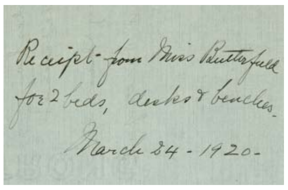
The evaluation was done on a low end machine to demonstrate that ability of the system to run on these kinds of machines.

6.1 H-IGT on DIBCO'11

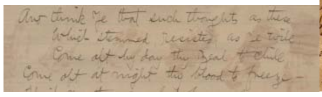
DIBCO’11 dataset consists of a collection of eight (8) handwritten documents and eight (8) printed documents with their respected Ground Truth (GT) documents. The original documents are shown below:


may 6. We have
Edyth Totten
ats) in the center
ally located. This
ill felt it was
money. [illegible]

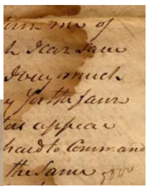
HW 1


Receipt from Miss Butterfield
for 2 beds, desks & benches.
March 24 - 1920.

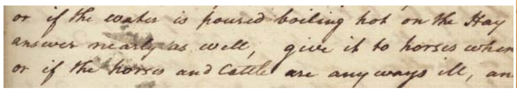
HW 2


And think ye that such thoughts as these
Which stunned resistes as ye twile
Come ot thy day the zeal to chile
Come ot at night the blood to freeze –

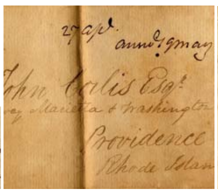
HW 3


[illegible] me of
h Dear Sam
very much
y for the Same
[illegible] appear
hard to Command
the Same [illegible]

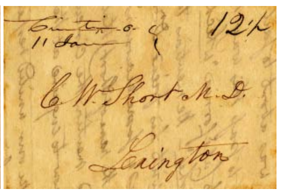
HW 4


or if the water is poured boiling hot on the Hay
answer nearly as well, give it to horses when
or if the horses and Cattle are any ways ill, an

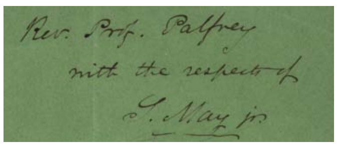
HW 5


27 apl. ansd. 9 may
John Carlis Esqr.
[illegible] Marietta & Washington
Providence
Rhode Islan

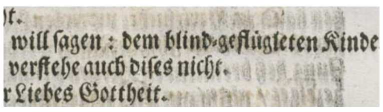
HW 6


[illegible]
11 Jan
12½
C. W. Short M. D.
Lexington

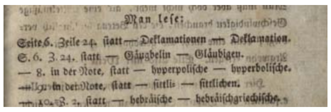
HW 7


Rev. Prof. Palfrey
with the respects of
S. May jr.

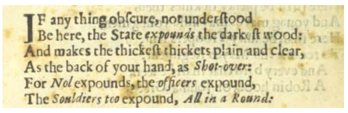
HW 8


ſt.
will ſagen : dem blind-geflügleten Kinde
verſtehe auch diſes nicht.
r Liebes Gottheit.

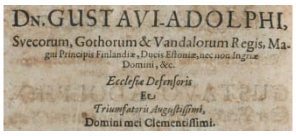
PR 1


Man leſe:
Seite 6. Zeile 24. ſtatt — Deklamationen — Deklamation.
S. 6. Z. 24. ſtatt — Gäugbelin — Gläubigen.
— 8. in der Note, ſtatt — hyperpoliſche — hyperboliſche.
— in der Note, ſtatt — ſittli- — ſittlichen.
— 3. 2. ſtatt — hebräiſche — hebräiſchgriechiſche.

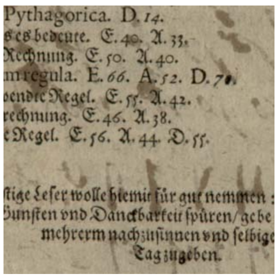
PR 2


IF any thing obſcure, not underſtood
Be here, the State *expounds* the darkeſt wood:
And makes the thickeſt thickets plain and clear,
As the back of your hand, as *Shot-over:*
For *Nol* expounds, the *officers* expound,
The *Souldiers too* expound, *All in a Round:*

PR 3

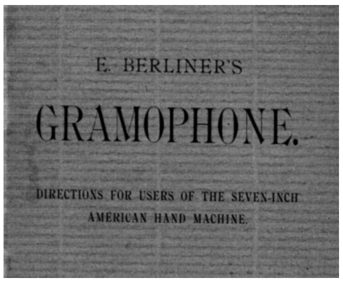

DN. GUSTAVI-ADOLPHI,
Svecorum, Gothorum & Vandalorum Regis, Magni Principis Finlandiæ, Ducis Eſtoniæ, nec non Ingriæ Domini, &c.
Eccleſiæ Defenſoris
Et
Triumfatoris Auguſtiſſimi,
Domini mei Clementiſſimi.

PR 4

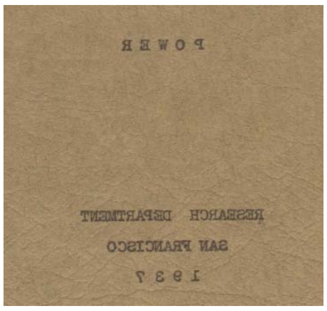

Pythagorica. D. *14.*
s es bedeute. E. 40. A. 33.
Rechnung. E. 50. A. 40.
m regula. E. *66.* A. *52.* D. 70.
oende Regel. E. 55. A. 42.
rechnung. E. 46. A. 38.
e Regel. E. 56. A. 44. D. 55.

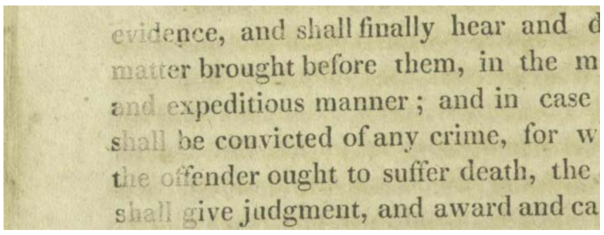
ſtige Leſer wolle hiemit für gut nemmen :
Gunſten vnd Danckbarkeit ſpüren/ gebe
mehrerm nachzuſinnen vnd ſelbige
Tag zugeben.

PR 5


E. BERLINER'S
GRAMOPHONE.
DIRECTIONS FOR USERS OF THE SEVEN-INCH
AMERICAN HAND MACHINE.

PR 6





evidence, and shall finally hear and d
matter brought before them, in the m
and expeditious manner; and in case
shall be convicted of any crime, for w
the offender ought to suffer death, the
shall give judgment, and award and ca

PR 7 PR 8

Figure 13. DIBCO’11 dataset

As described in 2.2.2 the H-IGT algorithm depends on three parameters; sensitivity k, window size n and termination criterion tc with default settings 2.5, 50 and 0.4 respectively as described in §3.2. Using these setting the algorithm was applied on the dataset and achieved an F-Measure of 81.3745%. In order to demonstrate that the system can find better settings, an SA session was performed on the dataset with the respective GT documents. The results are shown on the following table.

initial parameters	time (secs)	best parameters found	# of binarizations	Achieved F-Measure	Improvement (as percentage of the default F-Measure)
k=2,5 n=50 tc=0,4	1317	k=2,7291 n=65 tc=0,1251	2144	82,4114%	1,2743%

Table 2. H-IGT on DIBCO’11

It is straightforward that the system using SA improved the performance of the algorithm. Note that the session started from the default settings.. Another thing that is apparent from this test is that the designers of the algorithm have chosen the default settings carefully to work well on every type of document such as the DIBCO’11 dataset. The above test could very well lead to a change on the default settings of the algorithm since the improvement is considerable and the new settings are quite different than the default ones especially the termination criterion tc that leads to more iterations during the global thresholding phase.

6.2 FWLT on random document.

The second stage of evaluation was performed on the Fixed Window Local Thresholding (FWLT) algorithm. A single document was randomly chosen from the DIBCO’11 dataset. The reason a single document was chosen and not the entire dataset is because it would take considerably more time to explore the search space of the algorithm using Brute Force. Moreover, the scope of the test is to demonstrate the ability of SA to find very close to the best configuration settings in a fraction of the time needed to explore the entire search space. The document used is HW4 with its respective Ground Truth.

6.2.1 Brute Force

The results of using Brute Force to explore the entire search space are shown below:

time (secs)	**best parameters found**	**# of binarizations**	**Best F-Measure**
3556	k=79 w=15	15096	81,9694%

Table 3. Brute Force results on FWLT

The process took nearly an hour to finish although the range of the configuration settings of the algorithm was restricted for k to values between 50% and 100% and for w from 5 to 300 pixels. In addition to this, k was specified as integer although it could be specified as double. This gives only 51 values in the given range of the setting. As mentioned above the goal is to evaluate the performance of SA in comparison to Brute Force and not to find the best settings for the algorithm. As it is depicted in Figure 11 using k as double would not give us a better resylt on the best settings. It is obvious that a more complex algorithm with a bigger search space and a bigger dataset would require a much greater deal of time to explore the entire dataset. The graphical representation of the results is shown on Figure 6.

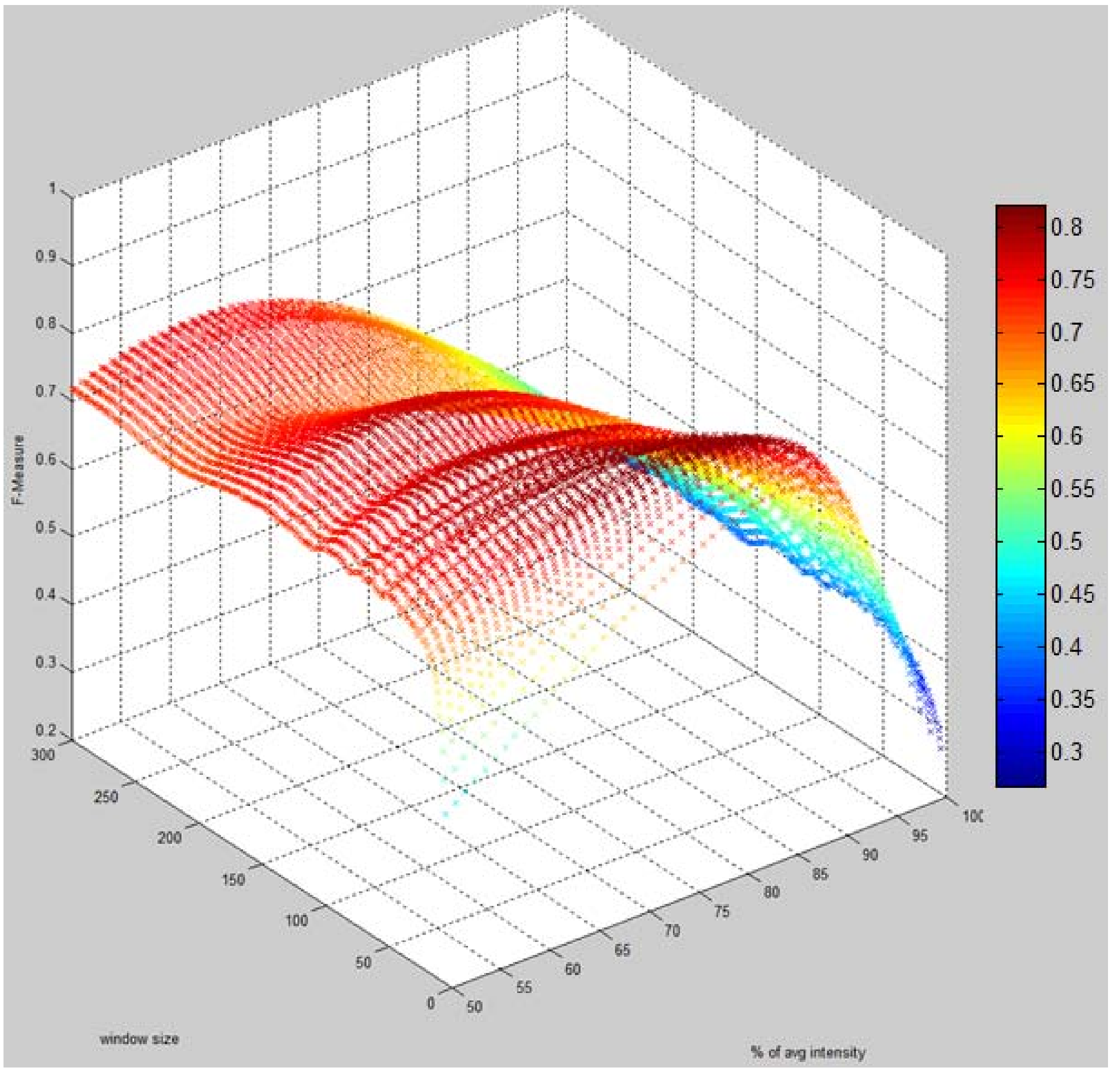


Figure 14. Brute Force Results

6.2.2 Simulated Annealing

In order to demonstrate the ability of SA to consistently find very good results, 100 SA sessions were performed on the same document using the algorithm’s default settings (k=82, w=50) as initial settings. The results are shown below:

#	time (secs)	best parameters found k w	# of binarizations	Achieved F-Measure	Difference (as percentage of the best F-Measure)
1	14	77 49	95	0,791566896	-3,43%
2	14	79 38	99	0,798197315	-2,62%
3	14	81 37	96	0,791214025	-3,47%
4	15	74 35	101	0,815865061	-0,47%
5	15	75 35	103	0,815575093	-0,50%
6	15	74 43	99	0,813115408	-0,80%
7	15	69 43	103	0,811832029	-0,96%
8	15	78 35	99	0,811630354	-0,98%
9	14	75 44	98	0,803878171	-1,93%
10	15	71 37	102	0,814529398	-0,63%
11	15	71 51	105	0,804840846	-1,81%
12	15	80 15	101	0,818815261	-0,11%
13	15	77 25	104	0,817384322	-0,28%
14	15	74 35	102	0,815865061	-0,47%
15	15	74 50	101	0,803003842	-2,04%
16	15	80 15	101	0,818815261	-0,11%
17	14	81 14	99	0,812979398	-0,82%
18	15	75 34	101	0,81372765	-0,73%
19	15	73 35	105	0,815648203	-0,49%
20	14	82 33	98	0,79077677	-3,53%
21	14	75 51	97	0,797672422	-2,69%
22	14	79 43	95	0,800313702	-2,36%
23	15	78 21	99	0,818958035	-0,09%
24	15	75 40	99	0,804407061	-1,86%
25	14	77 37	99	0,808043876	-1,42%
26	14	79 41	97	0,795937423	-2,90%
27	15	73 37	101	0,81442647	-0,64%
28	14	79 43	95	0,800313702	-2,36%
29	14	78 51	94	0,786401024	-4,06%
30	15	77 21	103	0,818905676	-0,10%
31	16	77 17	107	0,818193054	-0,18%
32	13	84 37	95	0,768963601	-6,19%
33	15	75 35	101	0,815575093	-0,50%
34	14	77 22	99	0,815256644	-0,54%
35	13	82 50	91	0,763929921	-6,80%
36	13	79 51	93	0,78122015	-4,69%
37	14	80 26	98	0,798251475	-2,62%
38	15	73 35	102	0,815648203	-0,49%
39	15	79 35	102	0,808816169	-1,33%
40	15	80 15	104	0,818815261	-0,11%
41	14	77 28	98	0,809463422	-1,25%
42	15	73 43	102	0,814177134	-0,67%
43	15	74 35	104	0,815865061	-0,47%
44	15	73 43	102	0,814177134	-0,67%
45	14	73 63	100	0,791564757	-3,43%
46	14	71 50	100	0,806810397	-1,57%
47	15	74 35	105	0,815865061	-0,47%
48	14	68 63	98	0,79685136	-2,79%
49	14	77 53	97	0,78165874	-4,64%
50	14	73 50	99	0,804136277	-1,90%
51	14	72 56	98	0,794077054	-3,13%
52	14	78 42	97	0,804724198	-1,83%
53	14	80 37	95	0,797424781	-2,72%
54	13	82 50	91	0,763929921	-6,80%
55	14	72 60	100	0,793290403	-3,22%
56	15	74 35	107	0,815865061	-0,47%
57	15	74 35	101	0,815865061	-0,47%
58	15	74 41	103	0,804515986	-1,85%
59	15	73 50	101	0,804136277	-1,90%
60	14	78 35	99	0,811630354	-0,98%
61	15	78 34	101	0,810484573	-1,12%
62	14	83 34	95	0,788950594	-3,75%
63	14	75 57	96	0,785282334	-4,20%
64	14	69 73	100	0,788482571	-3,81%

65	14	79 35	99	0,808816169	-1,33%
66	15	71 62	101	0,796406713	-2,84%
67	15	74 35	101	0,815865061	-0,47%
68	14	77 50	97	0,793871244	-3,15%
69	16	76 25	108	0,817359929	-0,28%
70	15	70 43	101	0,813161624	-0,80%
71	15	76 35	101	0,815104426	-0,56%
72	15	72 62	101	0,795988058	-2,89%
73	15	73 35	103	0,815648203	-0,49%
74	14	77 42	98	0,807352284	-1,51%
75	14	75 35	101	0,815575093	-0,50%
76	15	70 50	103	0,807485323	-1,49%
77	14	70 51	100	0,804799318	-1,82%
78	15	77 21	103	0,818905676	-0,10%
79	14	78 15	100	0,81906454	-0,08%
80	14	78 49	95	0,78772001	-3,90%
81	13	82 50	91	0,763929921	-6,80%
82	14	74 35	100	0,815865061	-0,47%
83	13	82 50	91	0,763929921	-6,80%
84	13	82 50	91	0,763929921	-6,80%
85	13	81 50	92	0,771342636	-5,90%
86	15	79 35	103	0,808816169	-1,33%
87	15	75 36	101	0,81315282	-0,80%
88	13	82 50	91	0,763929921	-6,80%
89	14	79 34	95	0,807643118	-1,47%
90	15	68 50	103	0,806344845	-1,63%
91	14	74 37	101	0,813643423	-0,74%
92	13	76 72	95	0,770462473	-6,01%
93	14	78 35	97	0,811630354	-0,98%
94	14	78 37	99	0,804848113	-1,81%
95	14	77 86	94	0,77094871	-5,95%
96	14	71 45	97	0,804495663	-1,85%
97	14	77 43	98	0,806753645	-1,58%
98	15	69 50	100	0,807657435	-1,47%
99	14	79 42	94	0,80098243	-2,28%
100	15	74 43	103	0,813115408	-0,80%

Table 4. Simulated Annealing on FWLT

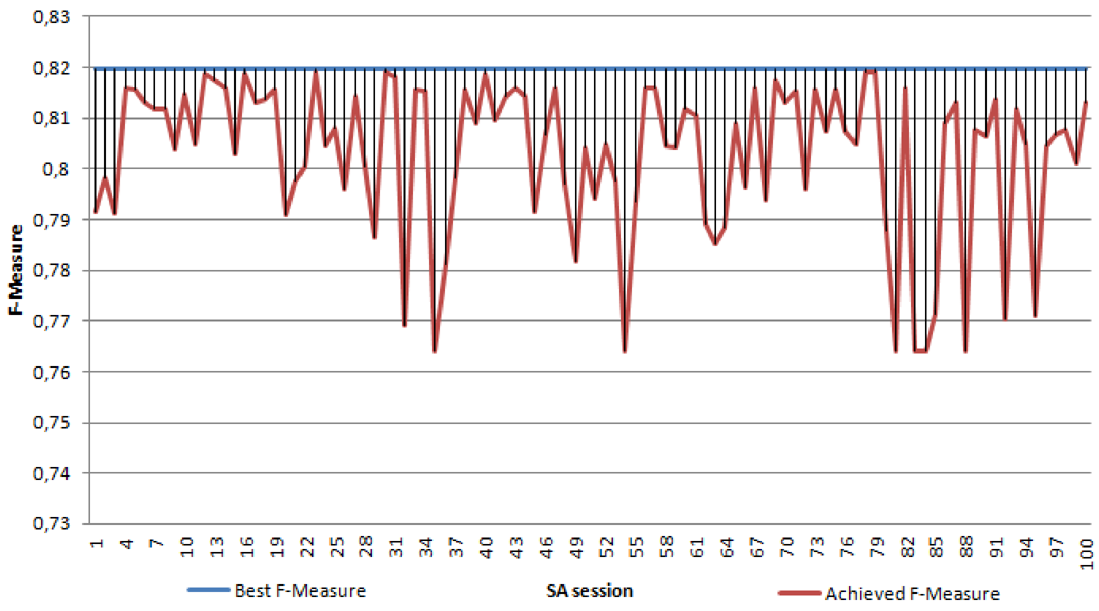


Figure 15. Simulated Annealing on FWLT

The results show that SA, as a probabilistic method, is unlikely to find the best overall solution especially in cases where the search space is enormous. However, the performance achieved is very close to the best settings. In particular the worst performance from SA came when the method could not improve from the default parameters that give a performance of 0,763929921. Figure 12 indicates that with a possibility of 89%, SA can find a solution within 5% of the best solution. In Figure 13 the produced binarized images of HW04 are shown for (a) the best settings found with BF and (b) the worse settings found with SA.

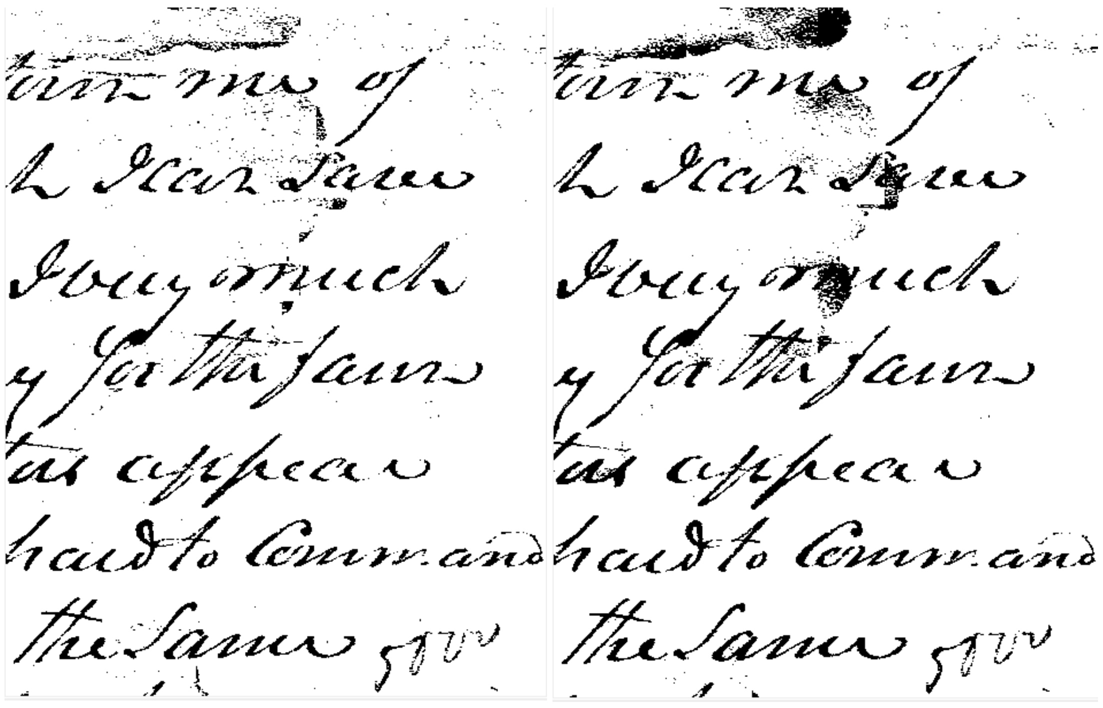


Figure 16. Best (a) and worse (b) performance of FWLT using SA

The great benefit comes from the time and number of binarizations required to achieve that result. On average it took less than 15 seconds for each session in comparison to 3556 seconds for the Brute Force; an improvement of 99,6%! In addition to this, the result was achieved with less than 100 binarizations on average a much smaller figure than Brute Force’s 15096. Taking into account that the search space was significantly reduced to expedite the evaluation, the benefit of the use of Simulated Annealing is straightforward. The corresponding graphical representation of four random sessions is shown below:

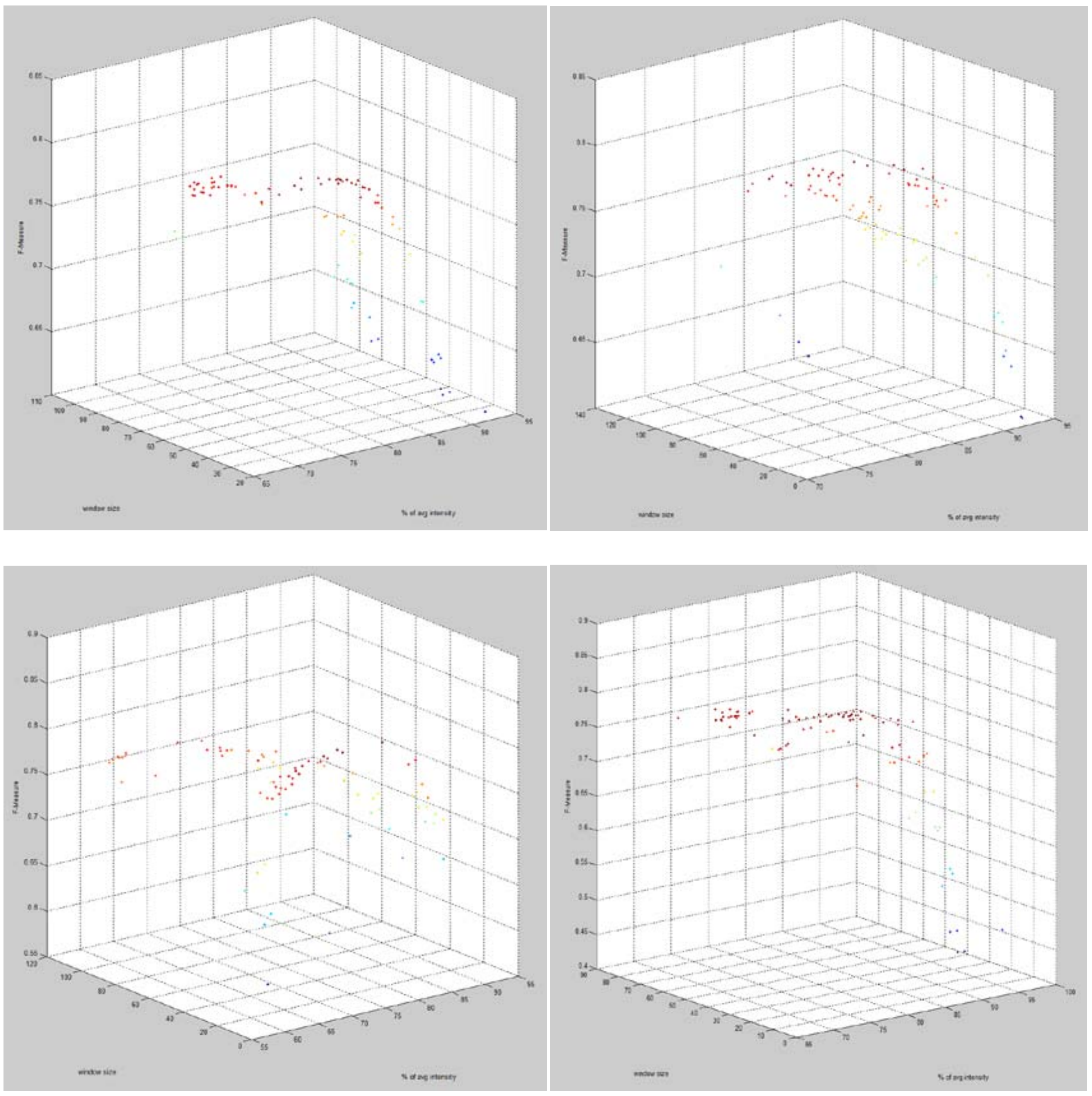


Figure 17. SA on FWLT random sessions

The difference between the above graphs and Figure 11 is visually evident. The SA process spends most of its time in the top area of the diagram within the area where the best performance of the algorithm is achieved. There are also scattered points in the periphery of that area indicating the effort of the SA process to avoid being locked in local minima.

6.3 Estimation of required GT images.

As mentioned in §1.2, when processing a collection of documents, only a small number of GT images is needed to run an SA session and then apply the settings found to the rest of the collection. Normally, if the GT images are not given, the user has to spend some time creating them with the aid of a common image manipulation application. This time spent is a good investment especially when large collections of images have to be processed. The question that arises is: “How many images are required?” In theory when a single image is used the SA process provides the best settings for that image only. If the image is chosen carefully to represent most if not all image degradations in the collection or the collection consists of images that uniformly have the same degradation issues then the settings found for that image will be good also for the collection. However, the more GT images you use the better the settings will fit for the collection.

In order to explore the number of images required, beyond which no significant improvement is made, a test was conducted using a subset of 70 images from the artificial database of Stathis P., Kavallieratou E. and Papamarkos N. [17] with its respective GT images. The subset was created taking images that suffer from the same type of degradation issues from the maximum intensity collection. The images taken are those created using background noise images 4, 5, 7, 9, 10, 11 and 15 from the collection. The main problems that these images suffer from are described in the following table:

Document	Problem(s)	Resolution
4	ink seepage, stains, strains	1188x889
5	stains, strains, stripes	1218x1405
7	uneven illumination, stains	1701x2340
9	uneven illumination, stains	2552x3509
10	background variation, stains	2552x3510
11	background variation, stains	2507x3510
15	background variation, stains	949x595

Table 5. Background noise images

An SA session was run using one random image at the beginning and subsequent sessions adding one image at a time. The images added to the first image where chosen properly in order to represent different background noise. The algorithm used was the FWLT. The settings found on each session were applied to the entire collection and the results obtained were compared to those of the respective SA session. The test terminated when the results converged and there was no significant improvement. Only six images were used overall for the SA sessions. The images used in the order they were used are 3_11.tif, 2_4.tif, 10_15.tif, 8_10.tif, 7_5.tif and 4_7.tif. There was no need to add an image with background noise 9 since the results converged earlier. The results are shown in the following table:

Documents Used	SA	Collection
1	0,999992773	0,864746979
2	0,991293602	0,931140304
3	0,976368426	0,942449619
4	0,95089029	0,940011457
5	0,960095792	0,95414573
6	0,96509773	0,948460735

Table 6. Estimation of required GT images - Results

The values depicted above are the F-Measure achieved. The convergence of the results is evident in the following diagram:

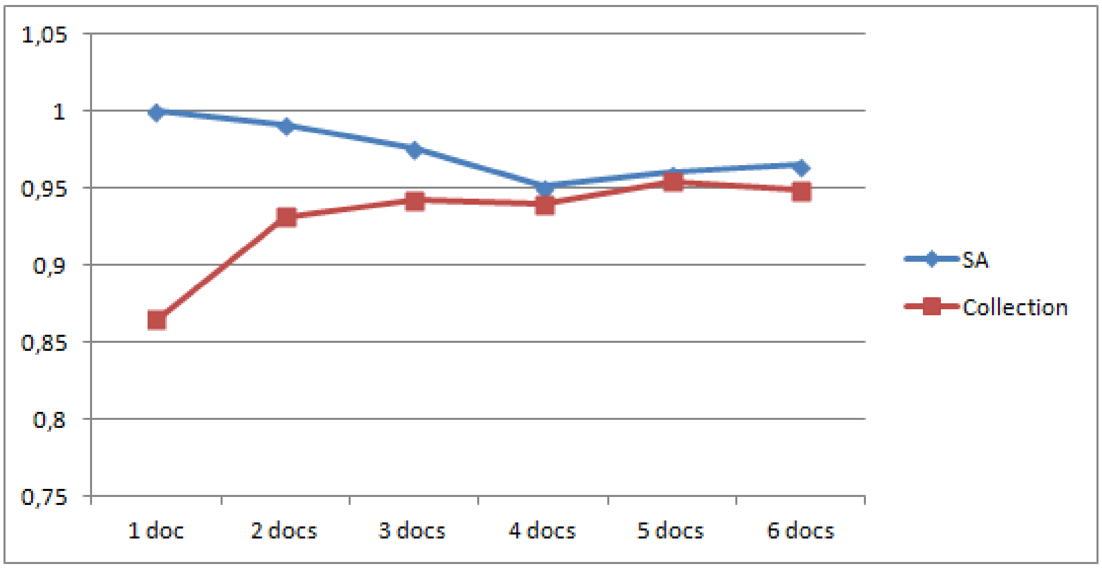


Figure 18. Estimation of required GT images - Diagram

After the third image, the settings found produce virtually the same results in the collection despite the fact that the convergence of the two lines is done in the fourth image. In the diagram above one can observe that the SA process achieved an F-Measure of almost 1 (0.999992773) for only one image. This indicates both the capabilities of the binarization algorithm (FWLT) and the SA process. As expected, as more images are added to the GT set, the SA process produces worse results for the GT set as the settings found now are not fit for each image independently but for all the images together. This is also reflected on the Collection line. As more images are added to the GT set, the settings found are more fit for the entire collection and not for individual images.

This test showed that for an almost uniform database as the one used, four (4) images are needed to be converted to GT images. The exact number of images needed for other collections depend on the specific characteristics of each collection but in any case, if chosen properly, this figure should be around what was found above.

6.4 Comparison of different fine tuning methodologies and binarization algorithms

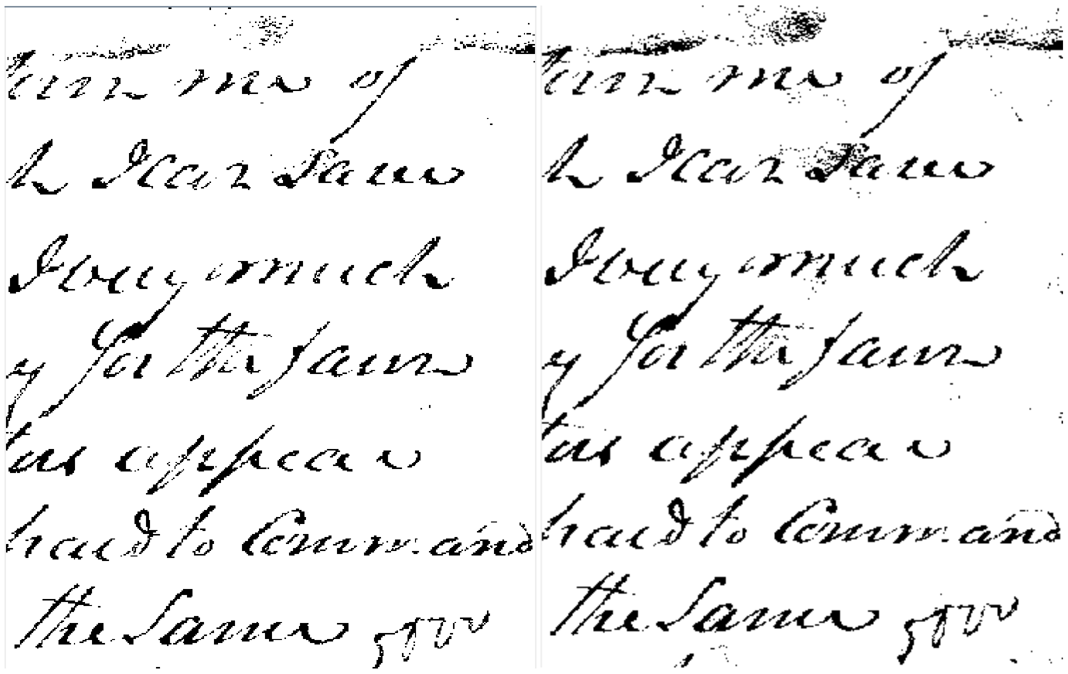
Apart from Simulated Annealing the system implements a feedback mechanism that facilitates the manual process of fine tuning a binarization algorithm as described in §5.2.3.3. A comparison test was performed for all three implemented algorithms using both SA and feedback mechanisms and comparing the results to the best possible results taken for each algorithm using Brute Force were applicable. In order to expedite the test, especially for Niblack's algorithm, only one image was used; the HW4 from DIBCO '11. This is the same image used in §6.2 and therefore the results obtained for FWLT could be reused. Another reason for using only one image was the fact that the feedback manual process gets complicated as a process when more images are added since it is harder for the user to assess the average results of a binarization in different images. This happens because, as mentioned in §1.1, different settings may work better for an image and worse for another image in comparison to previous settings and therefore the user cannot judge easily if an improvement was made. It is obvious that in case many images are required the SA process is the only way to go in order to fine tune an algorithm.

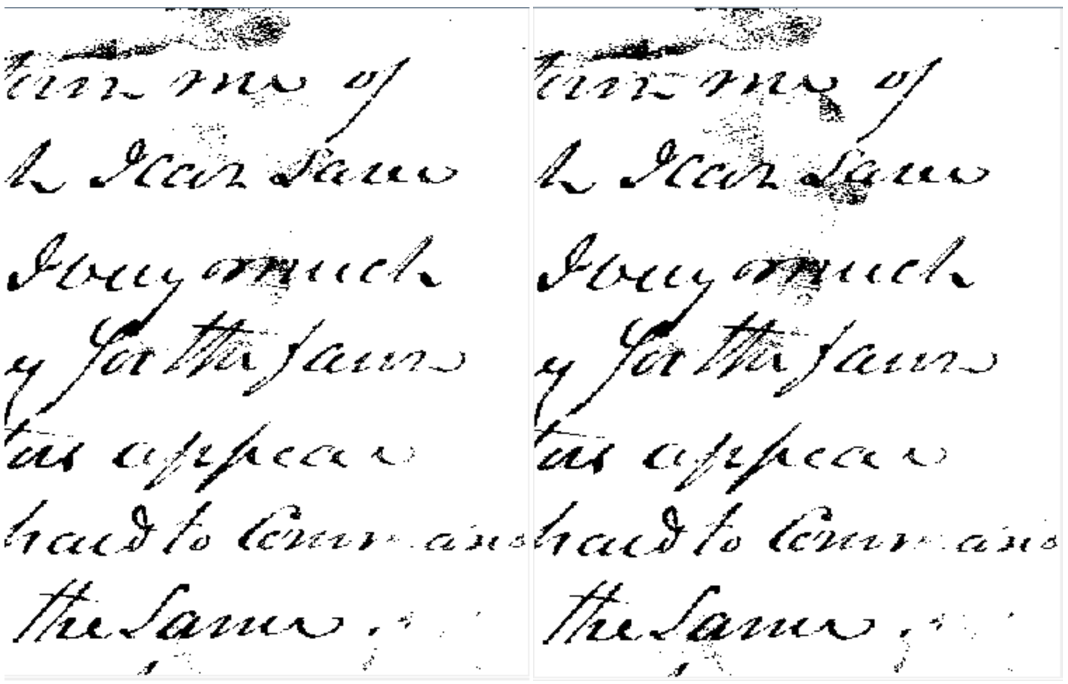
As mentioned above, a brute force was also run for comparison reasons. Unfortunately, this was not possible for Niblack since a BF would require roughly 21 days to be completed! The overall results are depicted in the following table:

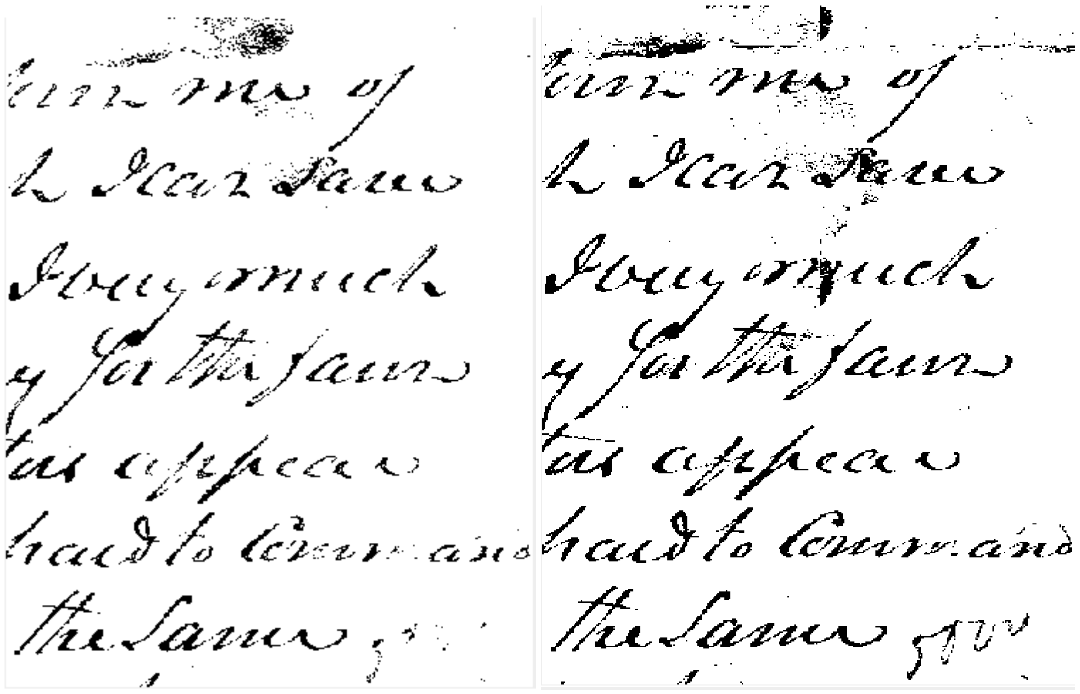
Methodology	Repetitions	initial parameters	Time (secs)	Best parameters found	# of binarizations	F-Measure	Difference (as percentage of the best F-Measure)	Precision	Recall
NIBLACK									
feedback	24	k=0.0 w=15	~5300	k=-1.7 w =100	24	0,768856669	-	0,944237	0,648421
SA	1	k=-1.0 w=95	18880	k=-1.4 w =92	99	0,790886118	-	0,868777	0,725813
brute force	-	-	~21 days	-	-	-	-	-	-
H-IGT									
feedback	25	k=2,5 w=50 tc=0,4	~750	k=1,0 w=100 tc=0,05	25	0,755667067	-1,65%	0,876994	0,66383
SA	1	k=2,5 w=50 tc=0,4	48	k=2,0 w=51 tc=0,24596588283675552	142	0,755155669	-1,71%	0,836134	0,688477
brute force	1	-	170244	k=1,0 w=62 tc=0,3	644930	0,768324504	-	0,793224	0,74494
FWLT									
feedback	24	t=82 w=50	~720	k=52 w=300	24	0,729080632	-11,05%	0,915628	0,605681
SA	1	t=82 w=50	19	t=75 w=35	102	0,815575093	-0,50%	0,860407	0,775184
brute force	1	-	3556	k=79 w=15	15096	0,819694245	-	0,877977	0,768668

Table 7. Summary of methodology and algorithm comparison

It is interesting to point out that, although the feedback methodology achieves worse or equal to the SA results, in terms of precision it is always the best method. During the feedback processes, the goal was to achieve the best image possible. This is subject to user interpretation. An effort was made to strike a balance between the text stroke and the amount of noise in the background. This led to texts thinner than the Ground Truth’s text and hence big False Negative (FN) and lower recall. On the other hand the noise was minimized in comparison to SA which accounts for lower FP and consequently better precision. These results are visually evident in the following figures:


(a)


(b)



(c)

Figure 19. Visual results of algorithm and methodology comparison. Left images for feedback and right images for SA. (a) for Niblack, (b) for H-IGT and (c) for FWLT

This comparison shows that a user with good knowledge of the way an algorithm works may reduce the search space considerably and find a good solution fast. However, even this fact requires from the user to invest time in trial and error process that takes a lot of time especially for algorithms like Niblack's that could take minutes to binarize an image. On the other hand, the SA process requires only the time to load the folders for the original and the GT images and then start the process. In addition to this, the time to create the GT images should be considered which in any case, as shown in §6.3, is small as it only requires a small number of images. Moreover, it is a task that has to be done once for every collection.

To sum up, if the task is to binarize large collections of images, the only feasible solution is to use the Simulated Annealing methodology.

7. Conclusion and future development.

In this thesis a semi-automatic system was developed that reduces the time and effort spent to fine tune binarization algorithms. The system incorporates a metaheuristic probabilistic optimization method called Simulated Annealing that is used to explore the search space of a binarization algorithm. This method is used in combinational optimization in order to find, with high probability, a good solution for a difficult multidimensional problem. A manual methodology for fine tuning binarization algorithms that is based on user feedback was also included. Three binarization algorithms were incorporated that were used for the evaluation of the system. Nevertheless, the system was designed so that it can be extended with the addition of new, user developed, algorithms using a trivial interface.

During the evaluation of the system its ability to apply SA on the image binarization problem was demonstrated successfully both in comparison to default and best algorithm settings that were found using a brute force method. In addition to this, the number of Ground Truth images required for an SA session to provide results applicable to the entire dataset was explored. This led to the conclusion that for a big dataset no more than 5 GT images are required. Finally, a comparison of both SA and user feedback is done for all algorithms in respect to a Brute Force approach were applicable.

The system is already expendable allowing for new algorithms to be installed by means of a simple interface. In the future this could also be done for the optimization method. Along with Simulated Annealing the user could create and install into the system custom methods. Then the user could choose the desired method and prior to starting the optimization session. Also some functionality could be implemented to assist the user in creating the Ground Truth images without having to open them in an image manipulation program. This could be done simply by clicking and inverting pixels on the image from foreground to background and vice versa.

8. Appendices

8.1 Appendix A: Algorithm developer manual

The application is designed to accommodate for the integration of new algorithms. This is done through the use of a service provider interface (spi) that allows the application to load new jar files and populate the algorithm list. A developer that desires to add a new algorithm needs to implement the ICSDAlgorithm interface. The only prerequisite for this implementation class is to have a zero-argument constructor and to implement the following methods:

- public BufferedImage **binarize**(BufferedImage input);

This method is the main method of the algorithm. The method provides the developer with a BufferedImage and expects a new BufferedImage that should be the binarized version of the image based on the current settings. The new image is displayed on the image display area. Inside this method the developer can implement the details of the new algorithm.

- public JPanel **getConfigurationPanel**();

The developer has to provide a JPanel with the visualization of the algorithm settings. If the algorithm has no settings then an empty JPanel should be returned.

- public String **getConfigurationDescription**();

The developer has to provide an abbreviated description of the current settings of the algorithm. For instance if the settings are window size and threshold the returned string could be “NiBlack_W30_T85”. This is the text that is appended to the image filename when a “Binarize All” action is performed.

- public ICSDAlgorithmParameter[] **getParameters()**;

The developer has to provide an array of all configuration attributes of the algorithm e.g Threshold.

- public boolean **setParameterValues(ICSDAlgorithmParameter[] parameters)**;

- public boolean **setParameterValues(Number[] newValues)**;
- public boolean **setParameterValue(int index, Number newValue, boolean updatePanel)**;

Whenever these methods are called, a new set of parameters or a single parameter is provided respectively.

- public Map<String, String> **getConfigurationMap**();

The developer has to provide a map of all the configuration setting names as keys along with the setting values as values. This map is used for logging.

- public void **resultOK**(); public void **resultUnder**(); public void **resultOver**();

These methods are called whenever a user feedback “OK”, “Under” or “Over” action is performed respectively. It is up to the algorithm developer to decide if there is any action to be taken. Normally for “Under” or “Over” the developer should suggest new settings towards the desired binarization level.

- public String **toString**();

This method should return the name of the algorithm displayed on the drop-down list.

Once the code is ready, the following steps should be taken to integrate the produced jar to the application:

- Copy the jar file into the lib folder.
- Open the applications binarization.jar file with winzip or winrar.
- Browse to the META-INF/services folder
- Open the edu.aegean.icsdm.binarization.algorithms.spi.ICSDAlgorithm file with a text editor.
- Enter a new line with the fully qualified name of the new algorithm’s class e.g. org.algorithms.NewAlgorithm. Save the file.
- Browse to the META-INF folder.

- Open the MANIFEST.INF file with a text editor.
- Enter the jar name in the class path e.g Class-Path: lib/AbsoluteLayout.jar, lib/NewAlgorithm.jar.
- Re-start the application.

8.2 Appendix B: Simulated Annealing Source Code

The class ICSDSimulatedAnnealing extends the class Thread. When the “start” button is pressed a new Simulated Annealing session is started and a new thread is spawned. The method run is called that contains most of the source code.

```
public void run() {
  //Initialize
  newStateCounter = 0;
  ICSDAlgorithmParameter[] parameters = algorithm.getParameters();
  int parameterSize = parameters.length;
  //Restrict the initial range so that we have a smaller neighborhood
  initRange = new double[parameterSize];
  for (int i = 0; i < parameterSize; i++) {
    double range = parameters[i].getMax().intValue() - parameters[i].getMin().intValue();
    initRange[i] = range / Math.pow(NB_MAX, 3);
    //Make sure that for integers the range is at least a lower limit
    if (parameters[i].getType() == ICSDAlgorithmParameterType.INTEGER) {
      if (MIN_INTEGER_RANGE > initRange[i]) {
        initRange[i] = MIN_INTEGER_RANGE;
      } else {
        initRange[i] = (int) initRange[i];
      }
    }
  }
  //Initialize state and counters
  initState = new ICSDState(algorithm, gTImages, origImages);
  newStateCounter++;
  // Current states and their respective energies
  ICSDState s = initState;
  ICSDEnergy e = s.getEnergy();

  // Initialize the best state
  S_BEST = initState;
  Date startTime = new Date();
  ICSDMainWindow.LOGGER.log(Level.FINE, startTime.toString());
  ICSDMainWindow.LOGGER.log(Level.FINE, "New Simulated Annealing session started with
        initial parameters:");
  printState(initState);
  ICSDState sNew;
  ICSDEnergy eNew;

  Random random = new Random();

  Number[] newValues = new Number[parameterSize];
```

```
for (int i = 0; i < parameterSize; ++i) {
  newValues[i] = s.getParameters()[i].getValue();
}

// While there is time left or there is improvement,
// and the optimum (minimum energy) is not achieved
for (int t = 0; t < MAX_ITERS && e.compareTo(ICSDEnergy.BEST_ENERGY) > 0; t++) {

  double temp = getTemperature(t);
  ICSDMainWindow.LOGGER.log(Level.INFO, "New Temperature: " + temp);

  boolean[] aBoundReached = new boolean[parameterSize];
  boolean[] bBoundReached = new boolean[parameterSize];

  for (int i = 0; i < parameterSize; ++i) {
    aBoundReached[i] = false;
    bBoundReached[i] = false;
  }
  // Pick some neighbor and compute its energy.
  int dim = 0;
  boolean foundLocalBest = false;
  parameters = algorithm.getParameters();
  // We loop for as long as there are more parameters and the local best state is not found
  for (int k = 1; dim < parameterSize; k++) {
    // We loop for as long as there are more parameters and the boundaries of the
    // current parameter are not reached
    for (; dim < parameterSize && !(aBoundReached[dim] && bBoundReached[dim]); dim++)
      {

      // Stop button pressed. Return current best state
      if (isInterrupted()) {
        ICSDMainWindow.LOGGER.log(Level.FINE, "SA execution was stopped by the user!!");
        ICSDMainWindow.LOGGER.log(Level.FINE, "Current Best State:");
        printBestState();
        Date endTime = new Date();
        ICSDMainWindow.LOGGER.log(Level.FINE, "Time Required: " + (endTime.getTime() -
      startTime.getTime()) / 1000 + " secs");
        for (ICSDSimulatedAnnealingEventListener listener : listeners) {
          listener.processInterrupted();
        }
        return;
      }

      // Find the range around the current value. The new neighbourhood
      // will be [value - halfRange, value + halfRange]
      double halfRange = (initRange[dim] * k * k * k) / 2;
```

```
// Find the left bound of the new neighborhood
double newABound = parameters[dim].getMin().doubleValue();
if (!aBoundReached[dim]) {
  newABound = s.getParameters()[dim].getValue().doubleValue() - halfRange;
  if (newABound < parameters[dim].getMin().doubleValue()) {
    newABound = parameters[dim].getMin().doubleValue();
    aBoundReached[dim] = true;
  }
}

// Find the right bound of the new neighbourhood
double newBBound = parameters[dim].getMax().doubleValue();
if (!bBoundReached[dim]) {
  newBBound = s.getParameters()[dim].getValue().doubleValue() + halfRange;
  if (newBBound > parameters[dim].getMax().doubleValue()) {
    newBBound = parameters[dim].getMax().doubleValue();
    bBoundReached[dim] = true;
  }
}

// Generate a new value at random from the neighborhood
double newValue = newABound + random.nextDouble() * (newBBound - newABound);
if (parameters[dim].getType() == ICSDAlgorithmParameterType.INTEGER) {
  newValues[dim] = (int) Math.round(newValue);
} else {
  newValues[dim] = Double.valueOf(newValue);
}

ICSDMainWindow.LOGGER.log(Level.INFO, "NEW Value for " +
 parameters[dim].getName() + ": " + newValues[dim]);

// Create a new state and compare it with the original state
algorithm.setParameterValues(newValues);
sNew = new ICSDState(algorithm, gTImages, origImages, temp);
newStateCounter++;
eNew = sNew.getEnergy();
// Diff of energy must be negative when we head to lower energy state. This means
// that the new F-Measure must be higher
double diffOfEnergy = e.getFMeasure() - eNew.getFMeasure();
ICSDMainWindow.LOGGER.log(Level.INFO, "\u0394\u0395: " + diffOfEnergy);
if (diffOfEnergy <= 0.0) {
  // We keep the state and replace the best if better
  s = sNew;
  e = eNew;
  if (eNew.compareTo(S_BEST.getEnergy()) < 0) {
    algorithm.setParameterValues(sNew.getParameters());
    S_BEST = new ICSDState(algorithm, gTImages, origImages, sNew.getTemp());
```

```
            newStateCounter++;
            ICSDMainWindow.LOGGER.log(Level.FINE, "New Best State found with F-MEASURE:
        " + S_BEST.getEnergy().getFMeasure());
          }
          foundLocalBest = true;
          continue; // to the next parameter
        } else {
          // We calculate the probability to keep the new state
          double probability = sNew.getTemp() == 0 ? 0
               : (sNew.getTemp() / T_start) * Math.exp(-diffOfEnergy / sNew.getTemp());
          if (probability > random.nextDouble()) {
            s = sNew;
            e = eNew;
            foundLocalBest = true;
            continue; // to the next parameter
          }
        }
      }
      if (foundLocalBest) {
        break; // Calculate new temperature. One iteration is done.
      }
    }
  }
  ICSDMainWindow.LOGGER.log(Level.FINE, "Optimum Parameters found!!!");
  printBestState();
  Date endTime = new Date();
  ICSDMainWindow.LOGGER.log(Level.FINE, "Time Required: " + (endTime.getTime() -
        startTime.getTime()) / 1000 + " secs");
  for (ICSDSimulatedAnnealingEventListener listener : listeners) {
    listener.processTerminated(true);
  }
}

/**
 * Get the temperature for the given iteration
 *
 * @param iter The current iteration
 */
private static double getTemperature(int iter) {
  return T_start * Math.pow(1 - Math.min(1, (double) iter / (MAX_ITERS - 1)), a);
}
```

References

[1] S. Vavilis, E. Kavallieratou, R. Paredes and K. Sotiropoulos, A Hybrid Binarization Technique for Document Images, 375 ed., Berlin: Springer Berlin Heidelberg, 2011, pp. 165-179.

[2] W. Niblack, An Introduction to Digital image processing, Prentice Hall, 1986, pp. 115-116.

[3] E. Badekas and N. Papamarkos, "Automatic Evaluation of Document Binarization Results".

[4] Y. Yitzhaky and E. Peli, "A Method for Objective Edge Detection Evaluation and Detector Parameter Selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, no. 8, pp. 1027-1033, 2003.

[5] M. Cheriet, R. F. Moghaddam and R. Hedjam, "A learning framework for parametric document binarization methods optimization".

[6] N. R. Howe, "Document Binarization with Automatic Parameter Tuning".

[7] J. He, Q. Do, A. C. Downton and J. H. Kim, "A Comparison of Binarization Methods for Historical Archive Documents," ICDAR’05, 2005.

[8] I. H. Osman and G. Laporte, "Metaheuristics: A bibliography," *Annals of Operational Research,* no. 63, pp. 413-623.

[9] S. Kirkpatrick, C. Gelatt and M. Vecchi, "Optimization by simulated Annealing," *Science,* no. 4598, pp. 671-680, 1983.

[10] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics,* vol. 6, no. 21, pp. 1087-1092, 1953.

[11] K. Prokopiou, "Thesis - Analysis, Design and Implementation of an Intelligent Interactive System for ruling-line removal from document images," Helenic Open University, Patra, 2012.

[12] W. Press, S. Teukolsky, W. Vettering and B. Flannery, "Numerical Recipes in C++. Example Book. The Art of Scientific Computing," *Cambridge University Press,* no. 10, 2002.

[13] "Java," [Online]. Available: http://www.java.com/en/.

[14] Oracle, "Java SE at a Glance," Oracle, [Online]. Available:

http://www.oracle.com/technetwork/java/javase/overview/index.html.

[15] Oracle, "Java Advanced Imaging," Oracle, [Online]. Available: http://www.oracle.com/technetwork/java/javase/tech/jai-142803.html.

[16] Netbeans, "NetBeans IDE 7.1.1," Netbeans, [Online]. Available: http://netbeans.org/.

[17] P. Stathis, E. Kavallieratou and N. Papamarkos, "An Evaluation Technique for Binarization Algorithms," *Journal of Universal Science,* vol. 14, no. 18, pp. 3011-3030, 2008.